

**UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE
CENTRO DE CIÊNCIAS EXATAS E DA TERRA
DEPARTAMENTO DE INFORMÁTICA E MATEMÁTICA APLICADA**

**X-Conf: Uma Infra-Estrutura de Gerenciamento e
Configuração para Serviços de Distribuição de Vídeo
Digital**

Fernando Luiz de Almeida

**Natal - RN
Novembro de 2005**

Livros Grátis

<http://www.livrosgratis.com.br>

Milhares de livros grátis para download.

Fernando Luiz de Almeida

**X-Conf: Uma Infra-Estrutura de Gerenciamento e
Configuração para Serviços de Distribuição de Vídeo
Digital**

Dissertação apresentada ao Programa de Pós-Graduação em Sistemas e Computação do Departamento de Informática e Matemática Aplicada da Universidade Federal do Rio Grande do Norte como parte dos requisitos necessários para a obtenção do título de Mestre em Sistemas e Computação.

Orientador:

Prof. Dr. Glêdson Elias da Silveira
UFPB

Co-orientador:

Prof. Dr. Guido Lemos
UFPB

**Natal – RN
Novembro de 2005**

*O progresso do mundo moderno nos oferece facilidade de conforto e prazer, de luxo e beleza
motivando-nos à conquista do sucesso.*

*A natureza é rica em sabedoria em abundância, mostrando-nos que Deus nos criou para usufruirmos
todas as coisas da vida.*

*Contudo, todo progresso só é positivo quando inclui em suas metas o sucesso do seu meio social.
Infelizes daqueles que, na ânsia de prosperar, atropelam os direitos dos outros, violando a lei da
integridade universal achando que cortam o caminho.*

Perceberão que o egoísmo e a ganância não são funcionais para se manter o fluxo da abundância.

*Divina a seu favor, pois a Natureza responderá com violência corretiva ensinando-os que só o
verdadeiro bem é capaz de proporcionar a prosperidade completa.*

*O caminho da verdadeira vitória é sempre árduo e cheio de surpresas desafiadoras que determinarão o
desenvolvimento de nossos potenciais inatos, garantindo a evolução do nosso espírito eterno.*

*A cada novo minuto você tem liberdade e responsabilidade de escolher para onde quer seguir, mas é
bom lembrar que TUDO NA VIDA TEM SEU PREÇO.*

Zíbia Milani Gaspareto

*Dedico este trabalho aos meus pais Jorge e Vera,
meus irmãos Leandro e Belisa e a minha querida
avó Albertina, pois eles são a maior razão desta
trajetória!*

Agradecimentos

Eu não teria chegado até aqui sem o amor, a paciência, o apoio e o incentivo dos meus pais Jorge de Almeida e Vera Lucia Bavia de Almeida, e dos meus irmãos Leandro Luiz de Almeida e Belisa Bavia de Almeida.

Este trabalho não seria realizado sem a oportunidade dada por meu orientador Prof. Dr. Glêdson Elias da Silveira, que, percebendo a importância do tema, sugeriu seu desenvolvimento. Seu desempenho profissional e sua capacidade de bem administrar os vários setores profissionais e pessoais da sua vida, tem servido de espelho para mim e para muitos outros pós-graduandos. Agradeço a ele a chance de estar aqui hoje. A parceria orientada e orientadora que serviu para fortalecer e preencher ainda mais os conhecimentos adquiridos no decorrer desta etapa.

Ao Prof. Dr. Guido Lemos de Souza, agradeço pelas oportunidades a mim concedidas em seus projetos de pesquisas e espaço em seus laboratórios para desenvolvimento do trabalho.

A Suelen Farias pelo apoio, motivação e ajuda nos momentos mais difíceis dessa caminhada, sempre muito compreensiva, paciente, me dando muita força para que eu pudesse atingir os objetivos e méritos conquistados.

Agradeço aos amigos do laboratório LAVID que contribuíram direta ou indiretamente para o desenvolvimento do trabalho, pela amizade e incentivo. De um modo especial agradeço a Marcos Vasconcelos (Marcovas), Gilberto Farias e Carlos Eduardo Batista (Bidu), pela indispensável contribuição prestada ao desenvolvimento deste trabalho.

Aos amigos e funcionários do Departamento de Informática da Universidade Federal do Rio Grande do Norte que de certa forma contribuíram para conclusão deste.

A todos que, de uma forma ou de outra, contribuíram para a realização deste trabalho, minha profunda gratidão e meu sincero reconhecimento.

Meu muito Obrigado!

Sumário

Capítulo 1.....	15
Introdução	15
1.1 – Motivação e Justificativa	16
1.2 – Objetivos	18
1.3 – Contribuições	19
1.4 – Estrutura da dissertação	20
Capítulo 2.....	21
Trabalhos Relacionados	21
2.1 – ALMADEM-VoD	21
2.2 – GLOVE – Global Vídeo Environment.....	22
2.3 – <i>Requestcasting</i>	25
2.4 – D-VoDs – <i>Distributed VoD System</i>	27
2.5 – DC-VoD System	29
2.6 – BufSharing – Buffer Sharing in Video-On-Demand Servers	30
2.7 – Comparação das propostas descritas.....	32
Capítulo 3.....	35
Arquitetura do Serviço de Distribuição de Vídeo	35
3.1 – Arquitetura de Alto Nível	35
3.2 – Plano de Coordenação	36
3.2.1 – Coordenador	38
3.3 – Considerações Finais	41
Capítulo 4.....	42
Linguagem de Configuração	42
4.1 – Especificação do Documento de Configuração.....	42
4.1.1 – NetNodes.....	44
4.1.2 – Blocks	46
4.1.3 – Connections.....	47

4.1.4 – Server	48
4.2 – Considerações Finais	51
Capítulo 5	52
Configuração e Gerenciamento do Serviço de Distribuição de Vídeo Digital	52
5.1 – Diagrama de Casos de Uso UML	52
5.2 – Visão Geral do Configurador.....	57
5.2.1 – Mapeamento dos nós na tela	58
5.2.2 – <i>Download</i> e <i>UpLoad</i> do Documento de Configuração.....	59
5.2.3 – Obter informação dos nós e estado operacional dos servidores	60
5.2.4 – Funcionalidade de zoom sobre o cenário	60
5.2.5 – Configuração dos Nós	61
5.2.6 – Configuração de blocos de endereço IP	68
5.2.6.1 – Mover Blocos de endereços IP para outro nó.....	71
5.2.7 – Configuração de uma conexão entre dois nós	72
5.2.8 – Configuração de Servidores.....	76
5.2.8.1 – Consulta manual do servidor <i>default</i>	79
5.2.8.2 – Configuração de um serviço	80
5.2.9 – Salvar informações	82
5.2.10 – Trocar o cenário do Configurador.....	83
5.2.11 – Barra de Menus <i>HELP</i> – Documento de Configuração.....	84
5.2.12 – Considerações Finais	85
Capítulo 6	86
Estudo de Caso.....	86
6.1 – Cenário 1: Transmissão SBRC e WRNP 2004.....	86
6.2 – Cenário 2: The 7th Annual SURA/ViDe Conference.....	95
6.3 – Considerações Finais	100
Capítulo 7	101
Considerações Finais.....	101

Referências Bibliográficas.....	103
---------------------------------	-----

Apêndices

Apêndice I	107
Noções Fundamentais de XML	107
Apêndice II	121
Documento de Configuração (Arquivo XML).....	121
Apêndice III	135
Documento de Validação (Arquivo XML Schema)	135

Lista de Figuras

Figura 1 – Sistema centralizado de distribuição de vídeo sob demanda ALMADEM.....	22
Figura 2 – Técnicas <i>Patching, Chaining e CVC</i>	24
Figura 3 – Árvore de encadeamento gerada pela CVC	25
Figura 4 – Arquitetura <i>Requestcasting</i>	26
Figura 5 – Recebimento de fluxo de vídeo pelo cliente	27
Figura 6 – Serviço de vídeo sob demanda distribuído.....	28
Figura 7 – Fluxo de dados do sistema de coordenação dinâmico	30
Figura 8 – Compartilhamento de <i>buffer</i> – ciclo corrente.....	31
Figura 9 – Grupo de clientes acessando o mesmo vídeo	32
Figura 10 – Arquitetura de Alto Nível.....	35
Figura 11 – Infra-estrutura do <i>X-Conf</i>	36
Figura 12 – Arquitetura do Coordenador	38
Figura 13 – Algoritmo de localização do cliente.....	40
Figura 14 – Representação do Elemento <i>NetDescr</i>	43
Figura 15 – Estrutura do elemento <i>netnodes</i>	44
Figura 16 – Descrição do elemento <i><netnodes></i>	45
Figura 17 – Representação do elemento <i><blocks></i>	46
Figura 18 – Representação do elemento <i><block></i>	46
Figura 19 - Descrição do elemento <i><blocks></i>	47
Figura 20 – Estrutura do elemento <i><connections></i>	47
Figura 21 – Representação do elemento <i><source></i>	48
Figura 22 – Descrição do elemento <i><connections></i>	48
Figura 23 – Representação do elemento <i><servers></i>	49
Figura 24 – Representação do elemento <i><pool></i>	49
Figura 25 – Representação do elemento <i><service></i>	49
Figura 26 – Descrição do elemento <i><servers></i>	50
Figura 27 – Diagrama dos elementos – Documento de configuração.....	50
Figura 28 – Diagrama UML do Configurador.....	52
Figura 29 – Diagrama UML da classe <i>NetNodes</i>	53
Figura 30 – Diagrama UML da classe <i>Connections</i>	54
Figura 31 – Diagrama UML da classe <i>Blocks</i>	55
Figura 32 – Diagrama UML da classe <i>Servers</i>	56
Figura 33 – Aplicação gráfica principal	58
Figura 34 – <i>Download e Upload</i> do Documento de Configuração	59
Figura 35 – Obter informações de um nó existente na rede de distribuição.	60
Figura 36 – Funcionalidade de zoom	61
Figura 37 – Inserção de um nó	62
Figura 38 – Interface de inserção de um novo nó.....	62
Figura 39 – Chamada da interface de remoção de um nó	63
Figura 40 – Confirmação de remoção de um nó	64
Figura 41 – Movendo blocos para outro nó.....	64
Figura 42 – Consulta de um nó na rede de distribuição	65
Figura 43 – Consulta a blocos alocados ao nó <i>Rio de Janeiro</i>	66
Figura 44 – Consulta as conexões existentes com o nó <i>Rio de Janeiro</i>	67
Figura 45 – Consulta a servidores associados ao nó <i>Rio de Janeiro</i>	68
Figura 46 – Inserção de um novo bloco de endereço IP	69
Figura 47 – Remoção de um bloco de endereço IP.....	70

Figura 48 – Substituição de um bloco de endereço IP.	71
Figura 49 – Movendo um bloco de endereço IP para outro nó	72
Figura 50 – Procedimento para inclusão de uma Conexão.	73
Figura 51 – Inclusão de uma conexão entre dois nós	74
Figura 52 – Remoção de uma conexão entre dois nós	74
Figura 53 – Procedimentos para editar uma conexão entre dois nós.....	75
Figura 54 – Edição de uma conexão ente dois nós.....	76
Figura 55 – Inserção de um servidor	76
Figura 56 – Inserção de um novo servidor	77
Figura 57 – Remoção de servidores	78
Figura 58 – Edição de um servidor.....	79
Figura 59 – Consulta manual do servidor <i>default</i>	80
Figura 60 – Informação manual do servidor <i>default</i>	80
Figura 61 – Inserção de um serviço.....	81
Figura 62 – Interface de inserção de um novo serviço	81
Figura 63 – Edição de um serviço	82
Figura 64 – Salvando as novas informações no documento de configuração	83
Figura 65 – Procedimentos para troca de cenário.....	83
Figura 66 – Interface que permite a troca de cenário utilizado pelo Configurador.....	84
Figura 67 – Barra de menus <i>Help</i>	84
Figura 68 – Arquitetura do Experimento SBRC/WRNP 2004.....	86
Figura 69 – Adição de um nó	87
Figura 70 – Inserção do nó SBRC	88
Figura 71 – Inserção de um servidor ao nó SBRC	88
Figura 72 – Inserção do servidor RS no documento de configuração.....	89
Figura 73 – Procedimentos para inserção de um serviço ao nó SBRC	89
Figura 74 – Inserção de serviços ao servidor <i>200.132.0.108 – SBRC</i>	90
Figura 75 – Inserindo uma conexão entre o nó <i>São Paulo e SBRC</i>	91
Figura 76 – Inserção de uma conexão entre os nós SP e SBRC.....	91
Figura 77 – Procedimentos utilizados na inserção de blocos ao nó <i>SBRC</i>	92
Figura 78 – Inserção de blocos ao nó <i>SBRC</i>	92
Figura 79 – Configuração SBRC/WRNP 2004	93
Figura 80 – Transmissão ao vivo do PoP – RS	94
Figura 81 – Arquitetura do experimento realizado.....	96
Figura 82 – Inserção do nó <i>Sura/Vide</i>	97
Figura 83 – Inserção do servidor <i>200.231.123.1 ao nó SuraVide</i>	97
Figura 84 – Inserção de uma conexão entre Brasil e EUA.....	98
Figura 85 – Inserção do bloco <i>128.61.199.0</i> ao nó <i>SuraVide</i>	98
Figura 86 – Configuração SuraVide 2005.....	99
Figura 87 – Representação do <i>link</i> Brasil – EUA.....	100

Lista de Tabelas

Tabela 1 – Comparativo das propostas estudadas	34
--	----

Resumo

O presente trabalho apresenta uma solução para a configuração e gerenciamento de serviços de distribuição de vídeo digital. Para tanto, é necessário conhecer a infra-estrutura de configuração mais favorável para distribuição do fluxo gerado entre os clientes. A solução proposta envolve a definição, descrição e identificação das informações que descrevem essa infra-estrutura, como, nós da rede, blocos de endereços, conexões existentes, taxa de conectividade, localização, servidores instalados e disponíveis, e tipos de serviços. Essas informações são descritas através de um documento XML (*eXtensible Markup Language*). No entanto, a recuperação dessas informações é uma atividade não trivial devido a pouca visibilidade de arquivos textuais em XML. Também desenvolvemos um esquema XML (*XML Schema*) que define abstratamente o vocabulário e restrições para os elementos que compõem o cenário que descreve a infra-estrutura de configuração. Para facilitar a manipulação dos dados referentes à configuração do serviço, foi desenvolvida uma ferramenta gráfica através da linguagem Java favorecendo portabilidade com diversas plataformas. Por fim, apresentamos alguns estudos de casos que ilustram a aplicabilidade da estratégia e ferramenta implementada. Os estudos de caso desenvolvidos utilizam a infra-estrutura de distribuição da rede de vídeo digital da RNP (Rede nacional de Ensino e Pesquisa) que se encontra em funcionamento no *backbone* da mesma.

Abstract

This work presents a solution for the configuration of video on demand services. To do so, it is necessary to know the best configuration infrastructure for the distribution of generated stream between clients. Our solution comprises of the definition, description and identification of information that describes this infrastructure, such as network nodes, address blocks, existing connections, connectivity rate, location, installed and available servers and service types. These pieces of information are described in an XML (eXtensible Markup Language) document. However, the retrieval of this information is a complex task due XML text files being barely visible. Also, we have developed an XML scheme that abstractly defines the vocabulary and restrictions for the elements that belong to the scenario which describes the configuration infrastructure. In order to facilitate the manipulation of the data regarding the service configuration, a graphical tool has been developed in Java, which allows for the portability between several platforms. Finally, we present some case studies that illustrate the applicability of the strategy and the developed tool. The case studies are based on the RNP's (National Education and Research Network) video on demand network infrastructure, which is working on the RNP's backbone.

Capítulo 1

Introdução

Com o grande crescimento das aplicações multimídia nos últimos anos, pode-se afirmar que as atividades humanas estão cada vez mais ligadas à comunicação. Percebe-se também um grande crescimento voltado para o oferecimento de serviços multimídia interativos, integrando voz, dados, vídeo e imagens em diferentes áreas de atuação. Hoje em dia, é cada vez mais comum que áreas como medicina, mercado financeiro, educação e entretenimento, utilizem essas aplicações.

Os desafios provenientes deste cenário de aplicações avançadas estimulam novas pesquisas, sobretudo em termos de comunicação de dados envolvendo instalações e operações de redes de alta velocidade, aplicações distribuídas, transmissão de dados multimídia, dentre outras. Com isso, tem-se dado grande importância ao desenvolvimento de software voltado para aplicações multimídia, principalmente no que tange mecanismos de distribuição e transmissão de dados.

Mecanismos de distribuição e transmissão de dados multimídia são imprescindíveis no contexto atual de serviços multimídia interativos. Esses mecanismos devem gerenciar a distribuição do fluxo de dados gerados pelas aplicações entre servidores e seus respectivos clientes. No entanto, quanto maior o tráfego de informações gerado na rede pelas aplicações, mais complexo torna-se o seu gerenciamento. No caso de aplicações envolvendo mídias contínuas (como áudio e vídeo) essa problemática é ainda mais crítica. Nessas aplicações, além da geração de grande volume de dados, existe uma exigência temporal para transmissão das mídias. Quando essa exigência não é respeitada, o atendimento do cliente pode ser sensivelmente comprometido.

O desenvolvimento de estratégias para distribuição eficiente de dados é uma tendência fortemente verificada nas aplicações que utilizam dados multimídia. Essas estratégias, além de identificar a rota de distribuição (geradores e consumidores de fluxo), devem considerar aspectos como proximidade e capacidade de transmissão das redes (*throughput*). Assim, o desenvolvimento de estratégias responsáveis pela configuração e otimização desse cenário são primordiais para o sucesso de aplicações e serviços que envolvam distribuição e transmissão de dados multimídia.

Nesse contexto, o presente trabalho visa apresentar uma infra-estrutura para a configuração de serviços de distribuição de vídeo digital, denominado *X-Conf (XML-Based Configuration)*. Esta infra-estrutura prove um melhor atendimento aos clientes que fazem uso do sistema. Para tanto, é necessário conhecer o ambiente de configuração mais favorável para distribuição do fluxo de dados entre os clientes.

A proposta *X-Conf* envolve a definição, descrição e identificação de informações, como nós da rede, blocos de endereços, conexões existentes, taxa de conectividade, localização geográfica, servidores instalados e disponíveis, bem como, os tipos de serviços, através de um documento de configuração. No entanto, a recuperação dessas informações é uma atividade não trivial devido a pouca visibilidade de arquivos textuais (arquivo XML). Nesse sentido, também desenvolvemos uma ferramenta gráfica para configuração e gerenciamento dessas informações relativas à rede de distribuição.

A estratégia implementada tem por base um documento XML contendo a descrição das informações da rede de distribuição. Para tanto foi desenvolvido também um esquema XML (*XML Schema*) que define abstratamente o vocabulário e restrições para os elementos que compõem o cenário que descreve o documento de configuração. A interface gráfica da aplicação para o gerenciamento do documento de configuração foi desenvolvida usando a tecnologia Java, favorecendo a sua portabilidade com diversas plataformas.

1.1 – Motivação e Justificativa

A RNP (Rede Nacional de Ensino e Pesquisa) conta com uma rede de distribuição de vídeo digital, onde são conectadas todas as Instituições Federais de Ensino Superior (IFES) indicadas pela Secretaria de Educação Superior do MEC e os Institutos de Pesquisa do MCT, além das agências e órgãos dos dois ministérios. Outras instituições de ensino e pesquisa como museus, hospitais, organizações não-governamentais, instituições de fomento e assemelhadas, podem fazer uso da rede acadêmica desde que se enquadrem no uso da mesma [27].

A RNP prevê a manutenção de uma rede acadêmica nacional que seja, ao mesmo tempo, uma infra-estrutura de alto desempenho para comunicação entre instituições de ensino e de pesquisa e um laboratório para testes e desenvolvimento de aplicações e tecnologias de rede avançadas. A rede de distribuição de vídeo da RNP conta com vários servidores distribuídos pelo país (servidores fontes e intermediários), bem como

ilhas de edição, máquinas codificadoras de vídeo digital e serviços disponíveis na rede de distribuição.

O Grupo de Trabalho de Vídeo Digital (GTVD), iniciativa da RNP, tem por objetivo implantar uma infra-estrutura baseada em sua rede de distribuição que ofereça suporte a aplicações envolvendo manipulação de vídeo digital. Esse esforço visa induzir o desenvolvimento de uma nova geração de aplicações de vídeo digital, explorando ao máximo o potencial de redes de alta velocidade no país. Outro papel importante deste grupo de trabalho é a elaboração de recomendações e padrões para a iniciativa Internet2 no Brasil, bem como a representação do país em fóruns de padronização internacionais. [12]. O GTVD faz uso de uma infra-estrutura de hardware e software para disponibilizar seus serviços. A infra-estrutura utilizada conta com equipamentos para captura de vídeo, digitalização, edição, armazenamento e transmissão de vídeo digital [13].

A idéia principal desse projeto é permitir que qualquer usuário possa ter acesso a determinados recursos, independente da aplicação de apresentação (*Windows Media Player, RealPlayer, VLC*, etc) ou da plataforma utilizada (*Windows, Linux*, etc). Com esse propósito, foram desenvolvidas várias aplicações e implantações de vários servidores distribuídos por todo país, geralmente nos *POPs* da própria rede da RNP.

Dentro deste contexto, foi proposto o desenvolvimento de uma aplicação piloto para o gerenciamento do ambiente de configuração para distribuição de vídeo digital. A idéia inicial era constituir uma infra-estrutura de configuração destinada à manipulação dos dados contidos em um documento de configuração, garantindo maior confiabilidade, flexibilidade e facilidade de uso. Esta infra-estrutura de configuração visa uma melhor distribuição de recursos face às necessidades dos usuários. O presente trabalho apresenta os resultados obtidos a partir da implementação dessa aplicação piloto. Além da manipulação de dados, foi implementada também uma aplicação gráfica, que possibilita a configuração e gerenciamento desses dados.

Durante a primeira fase do GTVD, o objetivo era definir uma solução para distribuição de fluxos gerados pelos serviços de distribuição de vídeo. A primeira solução encontrada utilizou a medição do retardo através do envio de mensagens ICMP¹, técnica bastante difundida através do programa *ping*. Para ter acesso ao serviço, o cliente utiliza uma página *web*² definida pelo próprio grupo de trabalho. Partindo deste princípio, eram enviadas mensagens ICMP dos servidores disponíveis até o cliente que

¹ ICMP – *Internet Control Message Protocol*

² www.lavid.ufbp.br

solicitou o vídeo. Com base no retardo apresentado pelo tempo de resposta dos servidores para o cliente, era definido o melhor servidor para atender a requisição, aquele que apresentasse o menor tempo de retardo ou menor tempo de resposta.

Após vários testes e experimentos realizados, constatou-se que esta solução não respondia satisfatoriamente as necessidades pretendidas, pois nem sempre, a medição desse retardo denotava o melhor servidor candidato, devido aos grandes congestionamentos da rede ou então pelo uso de *firewalls*³. Muitas vezes, o servidor com menor tempo de retardo, não era o mais próximo do cliente, comprometendo a distribuição do fluxo transmitido na rede.

No intuito de solucionar o problema descrito anteriormente, foi projetada uma solução baseada em uma infra-estrutura de configuração baseada em uma descrição topológica da rede, especificado em um documento XML que contém informações necessárias e primordiais para o funcionamento do sistema. Essa nova infra-estrutura além de proporcionar confiabilidade, tornou o sistema mais dinâmico, eficiente e preciso, garantindo que o servidor escolhido para atender uma determinada requisição, seja realmente o melhor ou mais próximo do cliente.

No entanto, a configuração do serviço é uma atividade difícil de ser realizada pelo administrador do sistema, pois envolve a manipulação direta do código XML. Levando em consideração essas limitações e tentando facilitar o uso do sistema, foi desenvolvida uma aplicação gráfica (interface visual) destinada à configuração e gerenciamento do serviço. Esta aplicação além de facilitar o uso pelo administrador, é responsável por qualquer manipulação que possa ser realizada no documento de configuração. As funcionalidades suportadas por essa aplicação são basicamente: a inclusão, alteração, exclusão e consulta aos dados contidos nesse documento.

1.2 – Objetivos

O objetivo geral deste trabalho é desenvolver uma infra-estrutura para a configuração e gerenciamento de serviços de distribuição e transmissão de vídeo digital no intuito de promover um melhor atendimento aos clientes que fazem uso do serviço. A solução desenvolvida utiliza o conhecimento do ambiente de configuração de modo a torná-lo mais favorável para distribuição do fluxo de dados entre os clientes.

³ *Firewalls* – mecanismo de segurança interposto entre a rede interna e a rede externa com a finalidade de liberar ou bloquear o acesso de computadores remotos aos serviços oferecidos dentro da rede corporativa.

Para atingir o objetivo geral, é necessário cumprir os seguintes objetivos específicos:

- Definição, descrição e identificação das informações topológicas da rede de distribuição. Para o contexto utilizado essas informações são as seguintes: nós da rede RNP, bem como sua localização geográfica, blocos de endereços (alocação de sub-redes), conexões existentes, taxa de conectividade, servidores instalados e disponíveis, tipos de serviços disponibilizados pelos servidores, tais como: transmissão de vídeo ao vivo e sob-demanda.
- Especificação de um esquema através da linguagem XML (*XML Schema*) que define abstratamente o vocabulário e restrições para os elementos que compõem o cenário do documento de configuração.
- Desenvolvimento de uma aplicação gráfica usando a tecnologia Java para configuração e gerenciamento das informações contidas no documento de configuração.
- Configuração e gerenciamento de serviços de distribuição e transmissão de vídeo digital no intuito de promover um melhor atendimento aos clientes que fazem uso do serviço.

1.3 – Contribuições

A realização do presente trabalho trouxe as seguintes contribuições:

- Definição e descrição de uma infra-estrutura de configuração através de uma especificação XML. É possível instanciar o modelo genérico (*XML Schema*) para situações específicas (documento XML);
- Definição e descrição de uma infra-estrutura de validação do documento de configuração XML, utilizando os mecanismos proporcionados pelo *XML Schema*;
- Desenvolvimento de uma aplicação gráfica responsável pela configuração e gerenciamento do serviço. A aplicação gráfica dispõe de forma visual e interativa as informações contidas no documento XML, de modo a facilitar a manipulação desses dados pelo administrador do sistema.

1.4 – Estrutura da dissertação

A dissertação está organizada da seguinte forma: O Capítulo 1 apresenta a introdução do trabalho. O Capítulo 2 descreve alguns trabalhos relacionados, procurando descrever o estado da arte em serviços de distribuição de vídeo, bem como ferramentas de configuração desses serviços. O Capítulo 3 apresenta a arquitetura e funcionalidades da solução implementada para a infra-estrutura de configuração. O Capítulo 4 descreve os elementos que compõem o documento de configuração. O Capítulo 5 descreve a aplicação piloto e o protótipo implementado. No Capítulo 6, serão apresentados dois cenários que descrevem um estudo de caso utilizando a proposta *X-Conf*. Finalmente, as considerações finais e algumas perspectivas serão apresentadas no Capítulo 7.

Capítulo 2

Trabalhos Relacionados

Atualmente existe uma grande variedade de trabalhos que envolvem problemas e soluções enfrentadas em aplicações de vídeo sob demanda. Isto comprova o grande interesse pelas aplicações multimídia através das redes de comunicação e será de grande utilização pelas aplicações em um futuro próximo. As duas estratégias usadas em sistemas de distribuição de vídeo são:

- **Centralizada** – sistemas que usam *hardware* e *software* específicos da camada de rede, não realizando funcionalidades de armazenamento intermediário dos dados;
- **Distribuída** – sistemas que utilizam a replicação dos dados nos servidores próximos dos usuários.

Segue uma descrição de algumas arquiteturas estudadas, que se correlacionam com o trabalho proposto, que conseqüentemente utilizam uma das estratégias acima mencionadas.

2.1 – ALMADEM-VoD

O projeto ALMADEM-VoD [34] foi desenvolvido pela Universidade Federal de Minas Gerais em parceria com outras universidades, tais como: UFRJ, PUC-Rio, UNICAMP, UFPE, CPqD-Telebrás, UCLA, Universidade de Massachusetts e Universidade de Maryland. Os servidores de vídeo são do tipo *True Video-on-Demand* (T-VoD), permitindo total controle do vídeo pelo cliente. Como diferencial, utiliza equipamentos de baixo custo, computadores comuns e sistema operacional *Linux*. Utiliza políticas de QoS avançadas permitindo a maximização de clientes atendidos simultaneamente.

Os servidores possuem suporte para 400 clientes no qual podem acessar o serviço através de computadores ou *set-top-boxes*⁴. Sua arquitetura é centralizada, e com isso, não permite o uso distribuído de recursos da rede, bem como configuração dinâmica de serviços. Como adota protocolos de controle próprios e usa uma aplicação

⁴ Equipamentos específicos para decodificação dos vídeos e conexão com aparelhos de televisão convencional

específica para exibição dos dados, torna sua arquitetura pouco flexível. Possui políticas de armazenamento de vídeo em disco, facilitando a recuperação dos mesmos (leitura). Os vídeos são lidos ciclicamente e em blocos de tamanho constantes, onde esse tamanho é definido pela capacidade de decodificação no cliente.

O sistema ainda utiliza algoritmos de controle de admissão para controlar o acesso dos usuários aos servidores. Quando há uma requisição a um vídeo, esses algoritmos verificam a existência de recursos suficientes, tanto de disco, como de rede para atender a requisição. A Figura 1 ilustra os componentes básicos do sistema vídeo sob demanda do projeto ALMADEM.

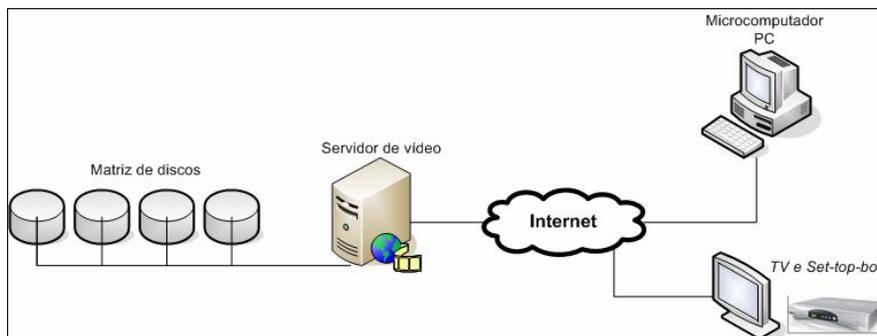


Figura 1 – Sistema centralizado de distribuição de vídeo sob demanda ALMADEM
Fonte: Adaptação da Figura 1 [34].

2.2 – GLOVE – Global Vídeo Environment

No contexto do projeto GLOVE [25] foi desenvolvido um sistema de vídeo sob demanda (VoD) pela UFRJ. Nesse sistema os clientes em atividade serão os responsáveis pela criação de uma memória *cache*, que é usada como recurso primário para os clientes subseqüentes. Esse sistema não limita o número de clientes que acessam o serviço simultaneamente, pois trabalha com um esquema de cooperativa, denominada *Cooperative Video Cache (CVC)*, onde os blocos de vídeo ficam armazenados temporariamente. Novas requisições poderão ser realizadas, sendo estas servidas pela CVC, aliviando a demanda dos servidores VoD.

A proposta GLOVE segue o modelo *peer-to-peer* e permite o uso de computadores de baixo custo. O modelo *peer-to-peer* colabora na resolução do problema da limitação de largura de banda encontrada nos servidores convencionais, permitindo o uso do *buffer* local dos clientes, ao invés de fazer uso dos recursos do servidor. A fim de agregar os benefícios das técnicas de transmissão de vídeo sob

demanda – *True-VoD*⁵, *Quasi-VoD*⁶ e *Near-VoD*⁷ – a proposta GLOVE analisa vários outros trabalhos acadêmicos, visando diminuir o uso da banda passante e ao mesmo tempo, gerar uma latência mínima de exibição do vídeo transmitido. Dentre eles destacam-se: *Batching*, *PiggyBacking*, *Patching* e *Chaining* [25].

No sistema *Batching* as requisições relativas a um determinado conteúdo são enfileiradas até que certa quantidade seja atingida, a partir do qual a transmissão se dá através de *multicast*.

A técnica *Piggybacking* consiste em fazer com que dois fluxos existentes com um mesmo conteúdo, porém defasado, sejam sincronizados, possibilitando a eliminação de um deles, acelerando assim a taxa de transmissão.

A técnica *Patching* parte do princípio de que os clientes têm capacidade de receber pelo menos o dobro do fluxo a ser exibido. O primeiro cliente que solicitar vídeo ao servidor recebe o fluxo *multicast* completo. Clientes posteriores que solicitarem vídeo passam a fazer parte do grupo de receptores do primeiro fluxo, sendo este armazenado em *buffer* local do mesmo. Ao mesmo tempo, um segundo fluxo (denominado remendo, do inglês *Patch*) é criado, contendo a parte compreendida entre o início do vídeo e a parte *bufferizada*.

A técnica de *Chaining* permite o reuso do vídeo através da criação de *buffers* correntes, no qual clientes ativos serão provedores de vídeos para outros clientes.

A técnica CVC surgiu baseada em uma abordagem híbrida entre *Chaining* e *Patching*, com ênfase no modelo *peer-to-peer*. Esta técnica trabalha com o *buffer* local dos clientes como componente de um sistema de memória global, capaz de prover conteúdos para os demais clientes. Com isso, os clientes tornam-se provedores preferenciais dos novos fluxos *multicast*, reduzindo o uso da banda e a demanda do servidor. Ao invés do servidor enviar *patches* para os novos clientes, os clientes em atividade se encarregarão de transmitir os blocos iniciais para eles.

A técnica CVC foi desenvolvida para ambientes de rede fechados, com estrutura homogênea. Necessita de largura de banda simétrica e abundante o suficiente para que o fluxo contínuo de vídeo seja efetivo. No que diz respeito à interatividade, foi

⁵*T-VoD* – Permite o controle total da programação ou sessão como avanço, reverso, pause e acesso aleatório.

⁶*Q-VoD* – Transmissão *multicast* para grupos de usuários.

⁷*N-VoD* – Exibição do vídeo em vários canais, com intervalos de tempos defasados, permitindo o avanço ou retrocesso em intervalos discretos no tempo.

implementado apenas recursos de *pause/recomeço*, visto que os clientes contam com pelo menos essa funcionalidade em um sistema VoD.

Essa técnica dispõe de um gerente centralizado para coordenar acessos correntes através dos *buffers* locais dos clientes. Este grupo representa um conjunto seqüencial de blocos de 128 KB definido no padrão MPEG. Para evitar o *overflow* nos *buffers*, a CVC realiza um controle através de cinco ponteiros, indicando posição de armazenamento, vídeos mais antigos, níveis de ocupação mínimo e máximo de *buffer*.

A Figura 2 ilustra as técnicas *Patching*, *Chaining* e *CVC*.

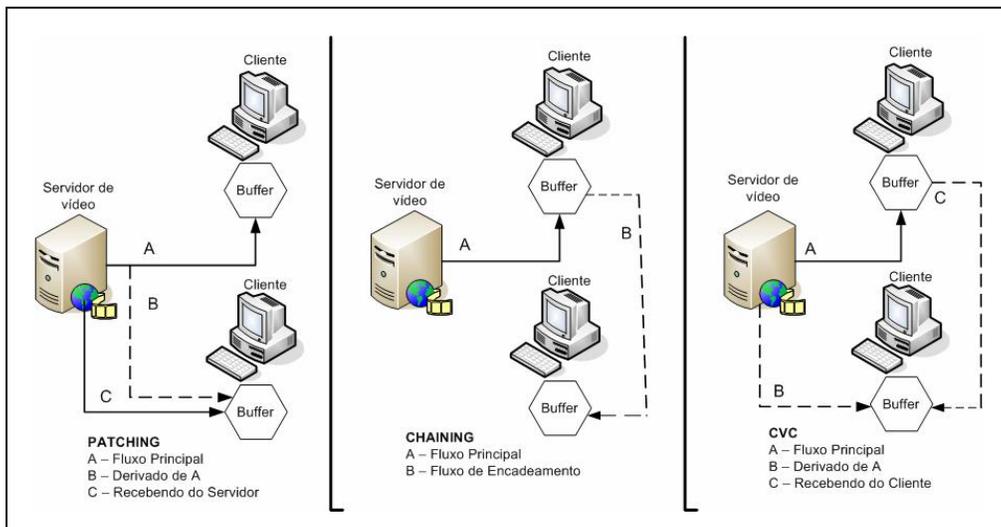


Figura 2 – Técnicas *Patching*, *Chaining* e *CVC*

Fonte: adaptação da Figura 1 [25].

O funcionamento dessa proposta baseia-se no atendimento das requisições dos clientes. Ao chegar a primeira requisição, o servidor inicia um novo fluxo para o cliente. Mesmo antes que o nível mínimo de buferização seja atingido, ele começa a exibição. Quando uma segunda requisição chegar, o gerente da CVC procura em sua tabela por um cliente, denominado provedor, que possua a parte inicial do vídeo em seu *buffer* local. Conforme os clientes conectam ao servidor, é gerada uma árvore de encadeamento que pode ser vista na Figura 3.

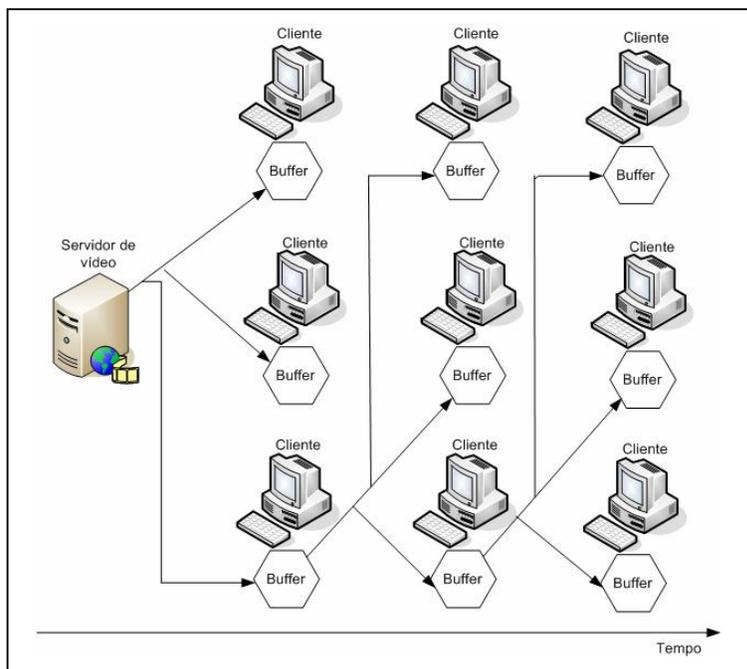


Figura 3 – Árvore de encadeamento gerada pela CVC
Fonte: adaptação da Figura 2 [25].

2.3 – *Requestcasting*

Os serviços de vídeo sob demanda têm como principais características a transmissão de vídeo de forma contínua, minimizando a utilização do canal, dar suporte às funções VCR (*rewing, pause e fast-forward*), bem como garantir o tempo mínimo de espera pelos clientes.

A proposta *Requestcasting* [24] é uma arquitetura de distribuição de vídeo sob demanda, no qual são combinados elementos de duas arquiteturas: de requisição e *broadcast*.

A arquitetura de requisição conecta clientes aos servidores através de canais bidirecionais. O cliente envia uma requisição para o servidor que a recebe, processa e transmite a resposta ao cliente.

Por outro lado, na arquitetura *broadcast*, os clientes são conectados ao servidor através de canais lógicos com largura de banda considerável. Os fluxos de vídeos são transmitidos independentemente das requisições dos clientes. Para receber os fluxos de vídeos transmitidos, os clientes devem apenas acessar o canal disponível pelo sistema.

A peça chave da arquitetura *Requestcasting* é o *Switching Board* o qual realiza o papel de interface entre o servidor e os clientes, fazendo a divisão entre as arquiteturas de requisição e *broadcast*. O *Switching Board* é conectado ao servidor através de banda larga (*high bandwidth*) e canais unidirecionais, enquanto os clientes são conectados

através de canais com banda básica (*low bandwidth*). A Figura 4 ilustra a arquitetura *Requestcasting*.

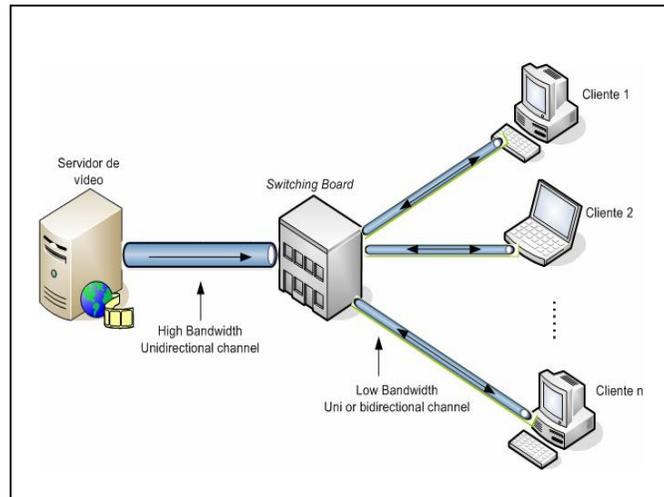


Figura 4 – Arquitetura *Requestcasting*
Fonte: Adaptação da Figura 1[24].

O servidor transmite vídeos contínuos em *broadcast* para o *Switching Board*, que retransmite ao cliente apenas o vídeo solicitado em *unicast*. Para diminuir o tempo de espera dos clientes em relação à comutação entre as duas bandas passantes (*servidor-switching-cliente*), foram implementados protocolos responsáveis em descrever a organização dos servidores de vídeos, o acesso aos canais de transmissão, além de fazer o controle de acesso dos clientes.

A banda passante é dividida em canais lógicos de mesmo tamanho. A capacidade de cada canal é definida com base na largura de banda e número de canais, enquanto os vídeos são divididos em blocos ou segmentos. Cada canal é responsável pela transmissão de um determinado bloco, o qual é substituído em uma ordem seqüencial através dos canais lógicos. O cliente aguarda o recebimento do primeiro segmento, sendo este armazenado em sua memória local. O *download* e a leitura deste segmento ocorrem de forma concorrente. Finalizada a recepção do primeiro segmento, o canal é comutado e a recepção do próximo segmento é iniciada. A troca de canal resulta na recepção de seguimentos maiores do que o anterior. Este procedimento se repete enquanto houver segmentos do vídeo a ser recebido pelo cliente. A Figura 5 ilustra um exemplo desse procedimento de troca de canais durante a recepção de diferentes segmentos de um determinado vídeo.

Para garantir continuidade de recepção, a taxa de *download* deve ser maior do que a taxa de consumo. Os canais utilizados requerem uma largura de banda igual a taxa de *download* [24].

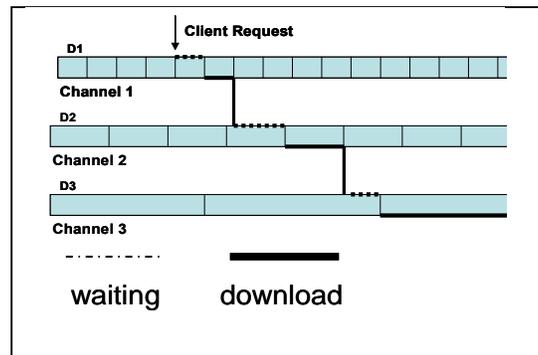


Figura 5 – Recebimento de fluxo de vídeo pelo cliente
Fonte: adaptação da Figura 2 [24].

As principais características da arquitetura *Requestcasting* são:

- Diminuição do uso da banda passante dos servidores, os quais transmitem vídeos contínuos em *broadcast* para um *Swiching Board* através de canais de banda larga;
- Protocolos de transferência de dados são responsáveis em fazer a comutação das bandas passantes, do *Swiching Board* para o cliente;
- Diminuição do tempo de espera pelos clientes;
- Aumento do número de clientes atendidos por vez;
- Permite o uso das funções VCR (*pause, rewind, fast-forward*).

2.4 – D-VoDs – *Distributed VoD System*

A arquitetura *Distributed VoD System* [19] propõe um sistema de vídeo sob demanda distribuído. Utiliza as técnicas de segmentação e *multicasting* para melhorar a capacidade dos sistemas de vídeo sob demanda. Consiste em uma hierarquia de arquivos multimídia conectados através de redes de alta velocidade (WANs⁸) e das redes metropolitanas (MANs⁹), como ilustrado na Figura 6.

Esta arquitetura consiste de servidores fontes de vídeo digital, responsáveis pelo armazenamento permanente dos vídeos. Um cenário real de um caso de uso pode ser visto em uma rede de TV a cabo com 450 canais disponíveis, com capacidade de 4,5 *Mbps*. Os responsáveis pelo armazenamento temporário (*bufferização*) dos segmentos

⁸ WANs – *Wide Area Networks*

⁹ MANs – *Metropolitan Area Networks*

dos vídeos são os *Set-top-boxes*. Seu principal objetivo é determinar o número de usuários suportados simultaneamente pelos servidores, minimizando a espera dos clientes. Quando uma requisição é aceita, o primeiro segmento do vídeo é transmitido. Após o término da transmissão do primeiro segmento, o segundo segmento será transmitido. Para assegurar continuidade dos vídeos, canais são reservados para a transmissão completa do vídeo.

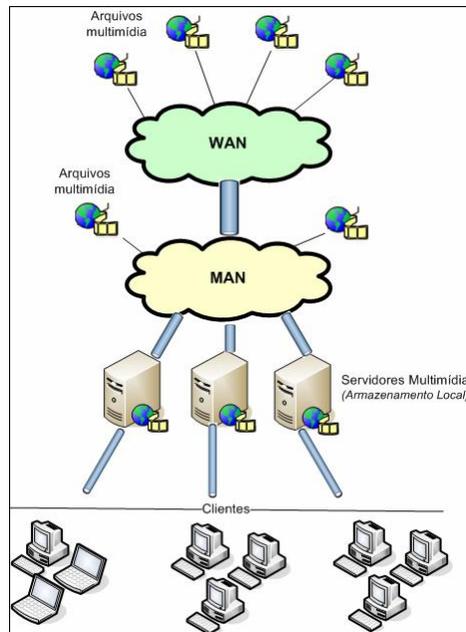


Figura 6 – Serviço de vídeo sob demanda distribuído
Fonte: adaptação da Figura 2[19].

Este trabalho também utiliza técnicas que controlam o número de requisições para um determinado vídeo. Por exemplo, um vídeo pode ser transmitido em *unicast*, quando o sistema constatar apenas uma requisição e ainda não possuir nenhum agendamento prévio para o mesmo. Quando mais de um cliente requisitar o mesmo vídeo ou este estiver agendado, o sistema assume a transmissão em *multicast* nos canais alocados para essa transmissão.

Quando uma requisição é aceita, o canal é reservado para a conexão, tornando disponível até que o vídeo seja inteiramente transmitido ao cliente. A rede de acesso que liga os servidores de distribuição (redes MANs) aos clientes é a mesma utilizada pelas TVs a cabo. As principais características são:

- Disponibilidade de armazenamento local;
- Controle total do canal alocado para cada transmissão;
- Segmentação dos vídeos em tamanhos fixos;

- Índice de popularidade igual para todos os vídeos disponíveis no sistema;
- Não possuem métodos que enfileiram requisições. Caso os canais estejam todos ocupados, as requisições são descartadas;
- Aumento de números de usuários atendidos simultaneamente;
- Baixo custo por usuário.

2.5 – DC-VoD System

A proposta *DC-VoD System* [23] é uma arquitetura de vídeo sob demanda destinada a prover um serviço de alta qualidade sobre redes WAN através da Internet. Esta arquitetura dispõe de servidores primários, denominados *Primary Server (PS)* por conter o vídeo original, ou servidores intermediários denominados *Cache Server (CS)*, no qual armazenam vídeos temporariamente, contendo assim uma cópia do vídeo. Esta arquitetura também apresenta um *Cache Agent (CA)*, responsável pela busca aos servidores que contém o vídeo requisitado, e por disponibilizar recursos de rede aos clientes.

Os servidores intermediários (CS) fazem o armazenamento dos vídeos requisitados aos servidores primários (PS). As funções de VCR (*fast-forward, pause e rewind*) são disponibilizadas aos clientes no momento em que o vídeo se encontra em *cache*, funcionando como uma cooperativa entre clientes. Para tornar o uso dos discos mais dinâmico, bem como garantir um balanceamento de carga das *CPUs*, mais de um *cache* (CS) pode ser instalado por LAN.

Um *Cache Agent (CA)* é um componente importante para o funcionamento dinâmico do sistema, pois é o responsável pela localização dos vídeos, bem como a distribuição dos recursos de rede para garantir qualidade ao serviço. Quando recebe uma requisição, um CA associa uma CS ao cliente dinamicamente. Isso é feito de acordo com a popularidade do vídeo armazenado (*cached*) e dos recursos disponíveis. Quando uma cópia do vídeo requisitado ainda não estiver armazenada (*cached*) de um servidor intermediário (CS), o agente (CA) é o responsável em liberar um CS para iniciar o *download* de um vídeo através de um PS, caso uma cópia do vídeo não exista na *cache* da CS. A Figura 7 ilustra o funcionamento completo da busca por vídeos no sistema de coordenação dinâmico.

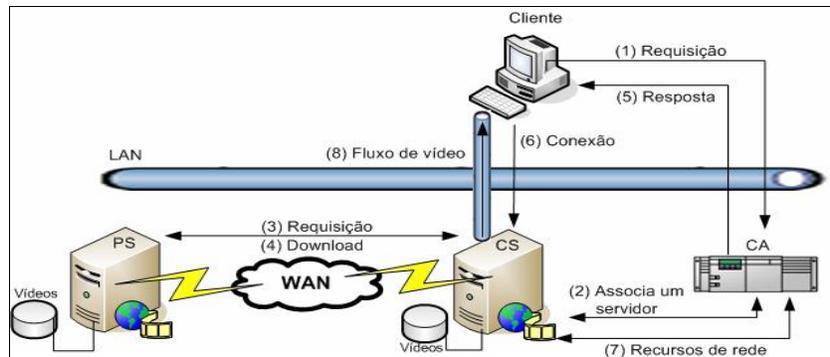


Figura 7 – Fluxo de dados do sistema de coordenação dinâmica
Fonte: adaptação da Figura 1[23].

Inicialmente o cliente faz uma requisição de um vídeo ao agente (CA). Este por sua vez busca por um servidor intermediário (CS) e o destina para atender a esta requisição. Se o CS não possuir uma cópia do vídeo requisitado, este solicitará ao servidor primário (PS). No momento em que uma cópia do vídeo estiver armazenada em um CS, o agente responde ao cliente com o endereço do CS que irá atendê-lo. O cliente estabelece uma conexão com um CS que solicita recursos de rede ao agente. Após receber esses recursos, o fluxo de vídeo começa a ser enviado ao cliente [23].

As principais vantagens da proposta apresentada são:

- Possui mecanismos que de acordo com o número de requisições a um vídeo determina a popularidade do mesmo;
- Possui gerenciamento de coordenação;
- Dispõe de mecanismos que balanceiam a carga do sistema e os recursos de rede, focando a alta qualidade do vídeo em questão.

2.6 – BufSharing – Buffer Sharing in Video-On-Demand Servers

Esta arquitetura apresenta uma técnica de compartilhamento de *buffers* que tem como foco principal fazer um balanceamento no uso dos discos dos servidores, bem como maximizar a desempenho dos servidores de Vídeo sob Demanda [30].

O principal objetivo dessa proposta é diminuir o número de acessos dos servidores aos discos. A idéia é que uma cópia dos vídeos mais populares se encontre armazenadas em memória *buffer*. No momento em que um cliente solicita um vídeo e este não se encontra disponível em *buffer*, o servidor faz uma busca em seu sistema de arquivos e o armazena em memória. Caso outro cliente solicite o mesmo vídeo, o servidor o atenderá com os dados que se encontram em sua memória *buffer* e não fará

outra busca no seu sistema de arquivos. Dessa forma, esta proposta garante que o vídeo é lido do disco apenas uma vez, e o sistema poderá suportar vários clientes simultaneamente.

A Figura 8 ilustra um exemplo de compartilhamento de *buffer*, em que dois clientes (C1 e C2) acessam a um determinado vídeo. A seta pontilhada ilustra um cliente sendo atendido pelo servidor, no qual este busca recursos em disco. A seta sólida representa um cliente que está sendo servido com os dados que se encontram em memória *buffer*. No corrente ciclo ilustrado na Figura 8, os clientes C1 e C2 consomem os blocos 8 e 3 respectivamente [30].

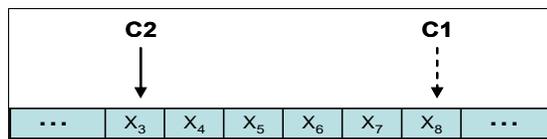


Figura 8 – Compartilhamento de *buffer* – ciclo corrente
Fonte: adaptação da Figura 1 [30].

A distância entre C1 e C2 é de quatro blocos. Quando os blocos referenciados por eles mudarem em função do tempo, essa distância permanece constante até que um deles termine seu processo.

A arquitetura proposta não suporta funções de VCR (*pause*, *resume*, *fast-forward* e *fast rewind*). Apresenta um esquema denominado *Controlled Buffer Sharing* (CBS) responsável por controlar o fluxo de dados (uso do *buffer* pelos usuários) e reduzir os requerimentos de disco pelos *frames* compartilhados [30].

O CBS permite que um determinado vídeo seja acessado simultânea e concorrentemente por diferentes clientes. Possui um algoritmo de gerenciamento de *buffers* que utiliza o conceito de distância entre os blocos. Responsabiliza-se também em definir o grupo no qual o vídeo pertence, e a seqüência em que os blocos devem aparecer, para não exceder a distância máxima permitida pelo sistema. Caso os blocos não estejam mais em *buffer*, estes deverão ser novamente requisitados ao disco, diminuindo consideravelmente o desempenho do sistema. Se um determinado vídeo estiver classificado como popular, este poderá ter um grande número de blocos consecutivos retidos em memória *buffer*. Dessa forma, uma simples requisição ao disco suportaria muitos clientes acessando o mesmo vídeo, aumentando consideravelmente o desempenho do sistema [30]. A Figura 9 ilustra vários clientes acessando o mesmo vídeo em *buffer*.

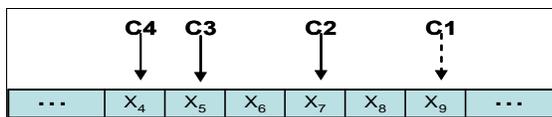


Figura 9 – Grupo de clientes acessando o mesmo vídeo
 Fonte: adaptação da Figura 3 [30].

2.7 – Comparação das propostas descritas

Um ponto crítico para distribuição de dados multimídia em arquiteturas distribuídas é a utilização de estratégias para distribuir e transmitir vídeos. Essas estratégias devem otimizar a distribuição de fluxos entre servidores e seus respectivos clientes, procurando compatibilizar a infra-estrutura de configuração disponível com as exigências da aplicação. Nesse sentido, nenhuma das propostas estudadas [34, 25, 24, 21, 23 e 30] contempla satisfatoriamente esse aspecto, pois não adotam políticas para otimizar a distribuição de vídeo digital, considerando a proximidade dos usuários e servidores de vídeo. Sendo assim, este trabalho discute mecanismos que permitam essa configuração da infra-estrutura de maneira eficiente e adequada às condições suportadas pela rede de distribuição de vídeo.

De acordo com as propostas estudadas, as arquiteturas distribuídas buscam diminuir a sobrecarga dos servidores fonte, armazenando vídeos nos servidores intermediários mais próximos dos clientes. Com isso, há uma melhor distribuição e otimização dos recursos pela rede. As propostas, *Requestcasting* e *D-VoDSys*, utilizam a técnica de transmissão *N-VoD*, no qual os vídeos são exibidos continuamente em vários canais, com intervalos de tempos defasados, permitindo que o cliente avance ou retroceda em intervalos discretos no tempo. Buscando agregar os benefícios das técnicas de transmissão de vídeo sob demanda (*T-VoD*, *N-VoD* e *Q-VoD*), a proposta *GLOVE* apresentou uma combinação das mesmas, para o melhor desempenho dos servidores de vídeo. As demais propostas fazem uso da técnica *T-VoD*, na qual os clientes tem controle total sobre a apresentação do vídeo em questão. O cliente pode escolher quando iniciar uma apresentação, bem como o que irá assistir, avançando ou retrocedendo para onde achar necessário.

Um fator relevante é em relação a banda passante necessária para que o cliente possa receber o vídeo desejado. As propostas *GLOVE*, *Requestcasting* e *ALMADEM*, por não transmitirem vídeos de alta qualidade, não necessitam de muitos recursos de banda passante. As propostas *D-VoDSys*, *DC-VoD System* e *BufSharing*, necessitam de

uma largura de banda considerável, enquanto que a arquitetura proposta se adapta a rede do cliente, transmitindo vídeos de acordo com a capacidade que o cliente possa receber.

No que tange a configuração do serviço, apenas o *D-VoDSys* e a arquitetura proposta apresentaram mecanismos de configuração dinâmica do serviço de distribuição. As demais possuem mecanismos de configuração estáticos, manipulado apenas pela administração do sistema. Por não possuírem mecanismos de configuração, as propostas GLOVE, ALMADEM e *DCSys*, não dispõem de recursos que controlam o acesso dos clientes ao sistema. *Requestcasting*, *DC-VoD System* e *BufSharing* dispõem desses recursos, e conseqüentemente apenas as redes cadastradas no serviço poderão fazer uso ao mesmo.

A proposta GLOVE segue o modelo *peer-to-peer*, colaborando assim na resolução do problema da limitação de largura de banda encontrada nos servidores convencionais. Para isso, faz uso dos *buffers* locais dos clientes em atividade no sistema, ao invés de fazer uso dos recursos dos servidores.

Flexibilidade é um fator que muitos materiais acadêmicos buscam para os serviços de distribuição de vídeo. Para isso, muitos trabalhos vêm sendo propostos em relação ao uso de *players* (disponíveis aos clientes), funções VCR, equipamentos de baixo custo para fornecer serviços a um grande número de clientes, configuração do serviço, entre outros. Nesse sentido, a arquitetura proposta apresenta alta flexibilidade, pois além das características acima mencionadas, permite ao cliente fazer uso de todos os *players* com suporte à transmissão HTTP (*Hyper Text Transfer Protocol*), independentemente da plataforma utilizada.

Para melhor visualização das principais diferenças das propostas estudadas, foi descrito em uma tabela um breve comparativo entre elas. Isso pode ser visualizado na Tabela 1.

Neste Capítulo abordamos algumas das soluções existentes atualmente, procurando visualizar o estado da arte em pesquisa e desenvolvimento para sistemas sob demanda que envolvem a transmissão e distribuição de vídeo digital.

Tabela 1 – Comparativo das propostas estudadas

Capítulo 3

Arquitetura do Serviço de Distribuição de Vídeo

Este capítulo introduz a arquitetura geral do serviço de distribuição de vídeo digital, e em seguida descreve as facilidades providas para configuração e gerenciamento deste serviço.

3.1 – Arquitetura de Alto Nível

A arquitetura proposta para o serviço de distribuição de vídeo baseia-se em uma estrutura hierárquica e distribuída, cujo principal objetivo é otimizar o tráfego na rede, reduzindo assim a utilização dos recursos de comunicação em *backbones*, redes regionais e redes institucionais. Como ilustrado na Figura 10, a arquitetura proposta é composta por três planos: publicação, coordenação, distribuição, e as aplicações de alto nível: de registro, visualização e gerenciamento.

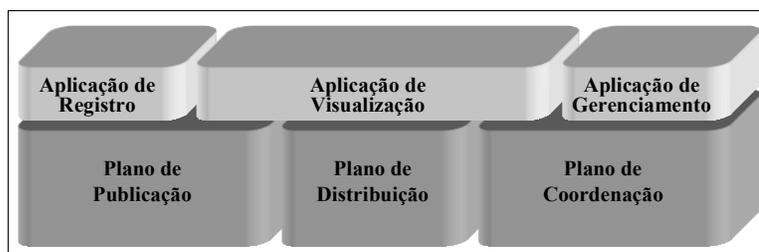


Figura 10 – Arquitetura de Alto Nível

Fonte: Figura 1 [4].

O *plano de publicação* é utilizado pelos provedores de conteúdo para publicar e armazenar réplicas dos seus respectivos vídeos. Caso o vídeo seja ao vivo, o agendamento da transmissão deve ser realizado, para tal, os provedores de conteúdo utilizam uma aplicação de registro. No processo de registro, cada vídeo é descrito por um conjunto de metadados, que podem ser posteriormente explorados por uma aplicação de visualização que incorpore facilidades de busca no plano de publicação. Além disso, o plano de publicação também é responsável por indicar à aplicação de visualização quais são as entidades do plano de distribuição que armazenam réplicas permanentes do vídeo requisitado pelo usuário.

O *plano de coordenação* é responsável por indicar à aplicação de visualização quais entidades do plano de distribuição representam a melhor alternativa para

recuperação de um determinado vídeo para um dado cliente. Para tal, as entidades do plano de coordenação coletam informações operacionais das diversas entidades do plano de distribuição, selecionando aquelas que estão correntemente operacionais e mais próximas do usuário. A adoção do conceito de proximidade dos usuários permite uma melhor utilização dos recursos da rede, uma vez que otimiza o tráfego gerado.

O *plano de distribuição* é responsável pela distribuição dos vídeos requisitados pelos usuários, respeitando as diretrizes sinalizadas pelo plano de coordenação. Neste processo, uma réplica do vídeo requisitado pelo usuário é recuperada de uma ou várias entidades do plano de distribuição e transportada para a estação do usuário.

Embora a arquitetura de alto nível seja estruturada na forma de planos independentes, internamente, os diversos planos podem trocar informações e compartilhar recursos. Por exemplo, os servidores que armazenam réplicas dos vídeos são utilizados pelos planos de publicação e distribuição [28].

3.2 – Plano de Coordenação

A infra-estrutura de configuração e gerenciamento proposta neste trabalho representa uma alternativa de solução para o plano de coordenação do serviço de distribuição de vídeo. O Plano de Coordenação busca sempre otimizar os recursos disponíveis na rede, bem como disponibilizar os melhores recursos ao atendimento do clientes. Como ilustrado na Figura 11, a infra-estrutura do *X-Conf* é composta por três entidades básicas: *configurador*, *coordenador* e *repositório de configuração*.

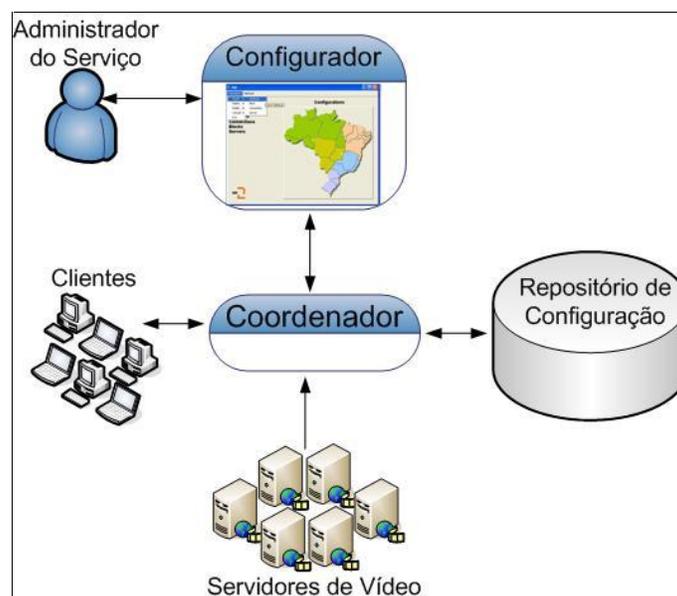


Figura 11 – Infra-estrutura do *X-Conf*

O repositório de configuração armazena as informações de configuração, que são organizadas em dois documentos: *configuração* e *validação*. O documento de configuração descreve informações topológicas da rede, identificando os nós que compõem a rede de distribuição, as redes alocadas em cada nó e as conexões existentes entre os nós. Além disso, o documento de configuração também identifica a localização dos servidores de vídeo na rede e os serviços oferecidos por esses servidores.

O documento de configuração é especificado através da linguagem XML, desenvolvida especificamente para representar as informações topológicas da rede de distribuição. A linguagem proposta é apresentada no Capítulo 4.

Por outro lado, o documento de validação define a estrutura da informação que pode ser incluída no documento de configuração. O documento de validação é especificado em XML *Schema* e define um vocabulário que, através de regras e restrições, sinaliza de forma não ambígua a sintaxe das informações contidas no documento de configuração.

O configurador é uma aplicação que permite ao administrador do serviço de distribuição de vídeo digital realizar a manipulação das informações de configuração, que são armazenadas e recuperadas do repositório de configuração. Para isso, o configurador solicita, através de uma conexão TCP (*Transmission Control Protocol*), uma cópia do documento de configuração ao coordenador. Após recuperar uma cópia do documento de configuração, usando o configurador, o administrador manipula localmente as informações de configuração. O configurador torna a manipulação das informações de configuração mais intuitiva, familiar e de fácil uso pelos administradores do sistema.

O coordenador tem como principal objetivo otimizar o uso dos recursos da rede. Para tal, explora o documento de configuração para identificar o servidor de vídeo mais próximo do cliente que fez a requisição ao serviço. Além disso, identifica a melhor rota entre o servidor fonte (que possui uma cópia permanente do vídeo) e os servidores intermediários (que possuem cópias temporárias do vídeo).

As informações contidas no documento de configuração são utilizadas pelo coordenador na construção de um grafo de conectividade, representando os servidores disponíveis na rede de distribuição. Após construir o grafo de conectividade, o coordenador utiliza o algoritmo de menor caminho, proposto por *Dijkstra* [5, 33 e 40], para identificar as melhores rotas entre clientes e servidores.

O algoritmo de *Dijkstra* calcula o menor custo entre vértices de um grafo. Inicialmente, é escolhido um vértice como raiz da busca, para posteriormente calcular o menor custo deste vértice aos demais vértices do grafo. Este algoritmo é simples e rápido, pois parte de uma estimativa inicial de custo e vai atualizando-a sucessivamente. Quando todos os vértices são percorridos, o valor associado a cada vértice representa o menor custo entre aquele vértice e o vértice raiz [40].

No documento de configuração, as conexões entre nós possuem um atributo que define a taxa de transmissão da conexão. Esse atributo é utilizado como peso das arestas do grafo de conectividade. Como o algoritmo de *Dijkstra* trabalha com a noção de custo, podemos concluir que, quanto maior a taxa de transmissão, menor é o custo da respectiva aresta. Desta forma, o custo de uma aresta é definido pelo inverso de sua taxa de transmissão.

3.2.1 – Coordenador

O Coordenador [31] é composto por dois componentes básicos: *gerente* e *monitor*. O gerente é responsável por determinar qual servidor irá atender a requisição do cliente, através do cálculo da melhor rota entre o cliente e o servidor. O monitor é responsável por verificar os estados dos servidores (ativos/inativos), permitindo que o gerente mantenha apenas rotas válidas. A Figura 12 ilustra a arquitetura básica do coordenador.

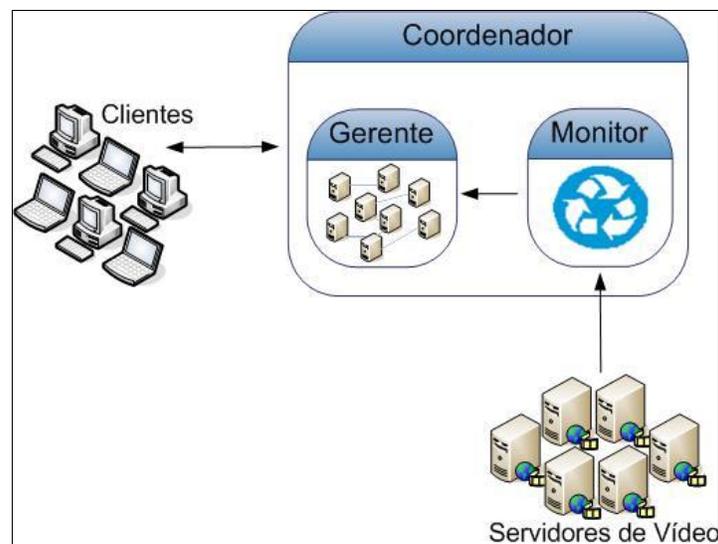


Figura 12 – Arquitetura do Coordenador

Quando o serviço é iniciado, o coordenador busca no repositório o documento de configuração e constrói o grafo de conectividade com base nas informações contidas nesse documento. Com o grafo totalmente construído, o algoritmo de *Dijkstra* é executado para calcular a melhor rota entre os clientes e servidores. O grafo é alterado sempre que existe a inclusão/exclusão de um servidor na rede de distribuição ou então após a mudança de estado (ativo/inativo) dos servidores existentes. Após qualquer modificação no grafo, o algoritmo de *Dijkstra* é novamente executado para calcular a melhor rota entre os clientes e servidores.

3.2.1.1 – Gerente

É responsável por escolher o servidor que irá atender a requisição de um cliente, bem como determinar a melhor rota entre o cliente que requisitou o vídeo e o servidor fonte, que contém o vídeo requisitado. De acordo com a arquitetura do serviço de distribuição de vídeo, o plano de distribuição é o responsável pela distribuição dos vídeos requisitados pelos clientes, respeitando as diretrizes sinalizadas pelo plano de coordenação. Nesse processo, uma réplica do vídeo requisitado pelo cliente é recuperada de uma ou várias entidades do plano de distribuição e transportada para a estação do cliente. A arquitetura do serviço de distribuição de vídeo atualmente suporta dois tipos de serviços: sob demanda – *D-VoD (Distributed Video on Demand)* e ao vivo – *D-Live (Distributed Live)*.

Para que o gerente possa determinar os melhores recursos disponíveis ao atendimento de uma requisição, é necessário identificar a localização (a rede a qual o cliente pertence) do cliente requisitante. No documento de configuração, cada rede cadastrada possui um atributo que descreve os seus blocos de endereços, que são identificados pelo endereço base do bloco e sua respectiva máscara. Esse atributo é utilizado para determinar a rede a qual o cliente pertence.

Para identificar a localização de um cliente, o gerente realiza a operação lógica *and* entre o endereço IP do cliente e a máscara de rede dos blocos contidos no documento de configuração. O resultado de cada operação lógica é comparado com os blocos de endereços descritos no documento de configuração. Se o resultado da operação é igual ao endereço base do bloco, então o cliente pertence à respectiva rede. A Figura 13 ilustra o procedimento para identificar a rede a qual o cliente pertence.

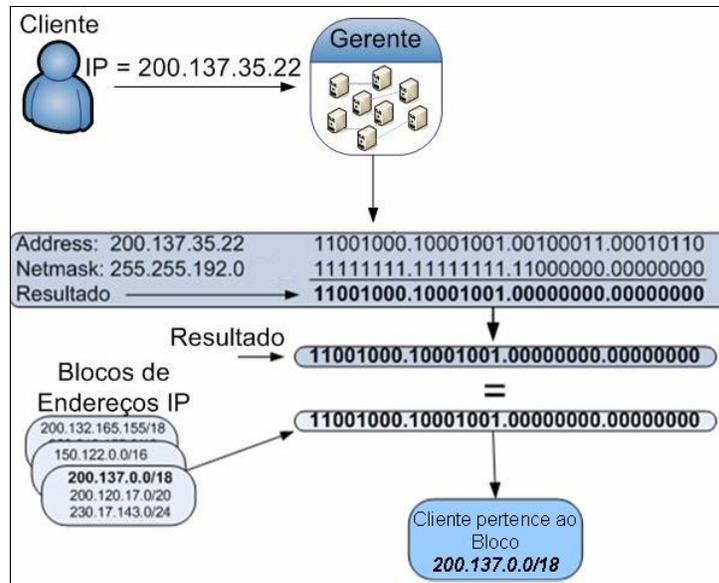


Figura 13 – Algoritmo de localização do cliente

Após identificar a localização do cliente, o gerente identifica a rota a ser adotada entre o servidor fonte (que mantém uma cópia permanente do vídeo requisitado) e o cliente, selecionando possíveis servidores intermediários (que podem manter cópias temporárias de blocos do vídeo requisitado) ao longo da rota. Por outro lado, o cliente requisita o vídeo desejado ao primeiro servidor intermediário da rota selecionada.

Alternativamente, o documento de configuração mantém a noção de servidor *default*, que é usado para atender clientes que não pertençam a redes (blocos de endereços) previamente cadastradas. Neste caso, o gerente identifica a rota entre o servidor fonte e o servidor *default*. Por outro lado, o cliente requisita o vídeo desejado diretamente ao servidor *default*, que é o primeiro servidor que compõe a rota.

3.2.1.2 – Monitor

É responsável por monitorar os servidores da rede, verificando o estado operacional (ativo/inativo) dos mesmos. Um servidor é dito ativo quando está operacional. Se um servidor torna-se inativo, o monitor sinaliza a falha ao gerente, que, por sua vez, atualiza o estado deste servidor no documento de configuração e conseqüentemente o retira do grafo de conectividade. Ao retornar para o estado ativo, o monitor sinaliza o evento ao gerente, que, por sua vez, atualiza o estado deste servidor no documento de configuração e o inclui novamente no grafo de conectividade.

Periodicamente, os servidores enviam uma mensagem de sinalização para o monitor. Esta mensagem identifica o servidor e informa a frequência de envio de

sinalizações. Após receber cada sinalização, o monitor atualiza o estado do respectivo servidor no documento de configuração.

Para cada servidor são guardadas duas informações: o estado atual e um contador. O estado atual corresponde ao estado em que se encontra o servidor em um determinado instante. Um servidor é dito *ativo*, se estiver em funcionamento e informando periodicamente seu estado ao monitor. O contador é uma variável associada a cada servidor disponível no sistema. Quando um servidor não envia a mensagem de sinalização ao monitor, esse contador é incrementado em um. O contador ficará armazenando a quantidade de vezes em que um determinado servidor não enviar sua mensagem de sinalização informando seu estado. No momento em que o contador associado ao servidor for igual a três, o monitor sinaliza ao coordenador o estado deste servidor para inativo, que conseqüentemente atualiza seu estado no documento de configuração e o retira do grafo de conectividade.

3.3 – Considerações Finais

Neste Capítulo abordamos a arquitetura de alto nível do sistema de distribuição de vídeo, cujo principal objetivo é otimizar o tráfego da rede, reduzindo assim a utilização dos recursos de comunicação em *backbones*, redes regionais e redes institucionais. Também foi descrito o Plano de Coordenação que é responsável por indicar a melhor alternativa para recuperação de um determinado vídeo para um dado cliente, visto que a adoção do conceito de proximidade dos usuários permite uma melhor utilização dos recursos da rede, uma vez que otimiza o tráfego gerado.

O Configurador e a Base de Configuração são as principais contribuições deste trabalho, que serão tratados nos Capítulos 4 e 5 respectivamente.

Capítulo 4

Linguagem de Configuração

Este capítulo descreve a estratégia adotada para o desenvolvimento do documento de configuração do serviço de distribuição de vídeo digital. Para tanto, é descrito um modelo de dados, utilizando a arquitetura XML, bem como sua validação através de um documento XML *Schema*.

4.1 – Especificação do Documento de Configuração

As informações topológicas da rede de distribuição de vídeo digital são especificadas em um documento de configuração que a descreve através de elementos e atributos. Esse documento é caracterizado por informações essenciais para o sucesso da estratégia utilizada para distribuição de vídeo. Essas informações são: nós da rede, conexões existentes entre nós, designação de sub-redes (blocos de endereços) aos nós da rede, servidores e serviços disponíveis. Com base nessas informações o coordenador constrói um grafo de conectividade entre servidores disponíveis no sistema. O documento de configuração reflete as características da rede de distribuição da RNP e possui toda estrutura definida em um documento de validação, especificado em XML *Schema*.

Nesse trabalho adotamos uma metodologia baseada em elementos estruturais UML [7] para representar todos os elementos que definem o documento de configuração, bem como o documento de validação. A Figura 14 ilustra um diagrama descrevendo o elemento principal `<netdescr>` juntamente com os elementos filhos – `<Netnodes>`, `<Connections>`, `<Blocks>`, `<Servers>`.

Toda informação contida no documento de configuração deve estar de acordo com as regras impostas pelo documento de validação, necessárias para garantir a consistência dos dados. O documento de validação especifica a estrutura e os tipos de informações que estarão contidos no documento de configuração.

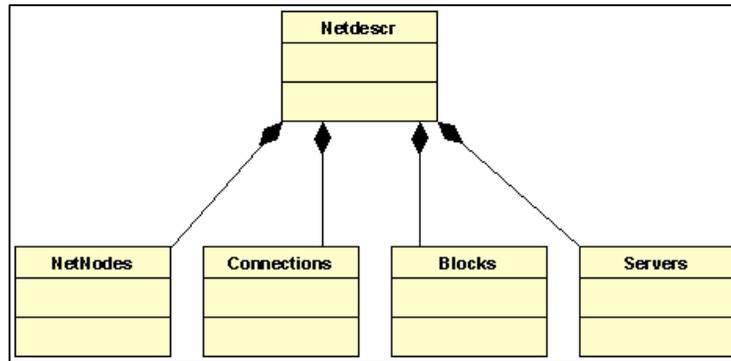


Figura 14 – Representação do Elemento *NetDescr*

Segue uma breve descrição de cada elemento que compõe a rede de distribuição de vídeo.

- **Descrição da rede** – descreve a rede de distribuição através de elementos e atributos que serão descritos abaixo. Foi definido como elemento principal e de partida do documento de configuração. É descrito pelo elemento `<netdescr>`;
- **Nós da rede** – descreve os nós existentes na rede de distribuição. São representados no documento de configuração pelo elemento `<netnodes>`;
- **Redes** – representam o conjunto de blocos de endereços IP alocados a um nó. São representados no documento de configuração pelo elemento `<blocks>`;
- **Conexões** – representam as conexões existentes entre nós e servidores. São representados no documento de configuração pelo elemento `<connections>`;
- **Servidores** – representam os servidores existentes no serviço de distribuição. São representados no documento de configuração pelo elemento `<servers>`. Um servidor pode ser um servidor fonte ou um servidor intermediário:
 - **Servidor fonte** – compõe o repositório permanente de vídeo digital, também chamado de servidor primário;
 - **Servidor intermediário** – possui um repositório temporário dos vídeos de maior audiência, também chamado de servidor secundário.

4.1.1 – NetNodes

O elemento `<netnodes>` descreve os nós (`node`) existentes na rede de distribuição de vídeo. Um nó possui seis atributos: `name`, `local`, `latitude`, `longitude`, `default` e `hasServer`. A Figura 15 ilustra o elemento `<netnode>` bem como seus atributos.

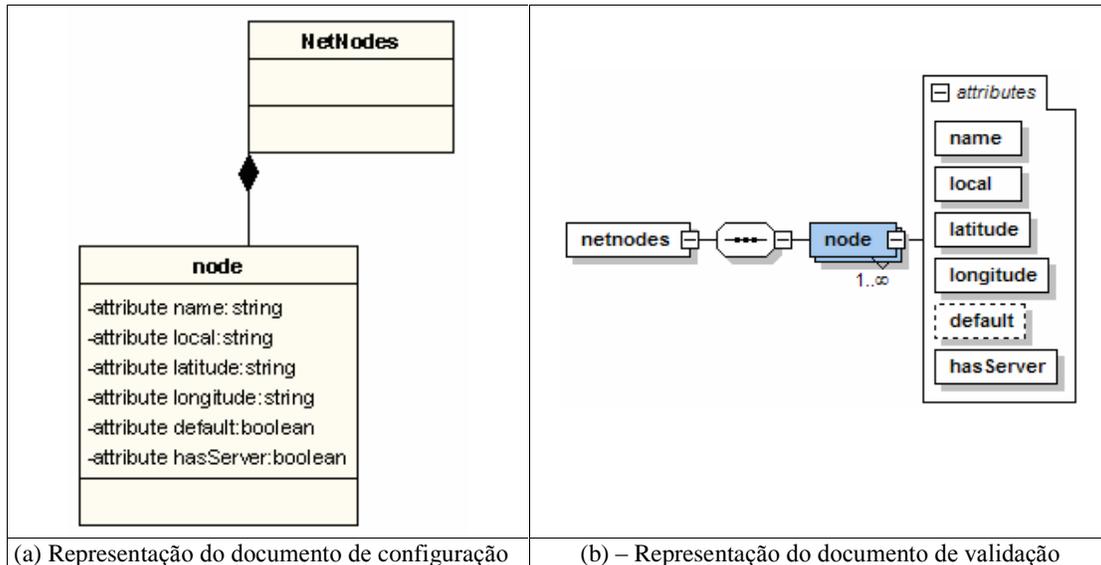


Figura 15 – Estrutura do elemento `netnodes`

A Figura 15(a) descreve os elementos contidos no documento de configuração, enquanto a Figura 15(b) os elementos contidos no documento de validação. Abaixo segue uma pequena descrição dos atributos contidos no elemento `<node>`:

- ✓ **name:** Este atributo define o nome do nó na rede de distribuição. Geralmente identifica o estado ou a instituição onde se encontra fisicamente esse nó;
- ✓ **local:** Descreve a localização física do nó na rede de distribuição, geralmente o nome da cidade onde se encontra;
- ✓ **latitude:** Representa o sistema de coordenadas utilizado para o mapeamento visual dos elementos na interface gráfica. A latitude é a distância ao Equador medida ao longo do meridiano de *Greenwich*. Esta distância é medida em graus, podendo variar entre 0° e 90° para Norte ou para Sul. São calculados automaticamente pela aplicação. Vale ressaltar que esta informação é armazenada no documento de configuração através de graus decimais¹⁰,

¹⁰ Graus decimais – Graus (hora), minuto(s) e segundo(s) convertidos para segundos (Ex: 30° 20' 55'' = 30,34869444).

sendo esta convertida para coordenadas geográficas¹¹, permitindo uma melhor visualização pelo administrador;

- ✓ **longitude:** Representa o sistema de coordenadas utilizado para o mapeamento visual dos elementos na interface gráfica. A longitude é a distância ao meridiano de *Greenwich* medida ao longo do Equador. Esta distância é medida em graus, podendo variar entre 0° e 180° para Leste ou para Oeste. Essas informações são calculadas automaticamente pela aplicação. Esta informação semelhantemente a latitude, é armazenada no documento de configuração através de graus decimais, sendo esta convertida para coordenadas geográficas no momento de sua representação;
- ✓ **hasServer:** Este atributo descreve se o nó possui servidor de vídeo instalado nele;
- ✓ **default:** O atributo *default* indica ao coordenador se o nó é servidor *default*. Como indicado anteriormente, um servidor *default* é usado para atender clientes cujos endereços não pertencem aos blocos de endereços cadastrados no documento de configuração.

A Figura 16 ilustra parte do documento de configuração que descreve os nós contidos na rede de distribuição da RNP. O documento completo pode ser visualizado no Apêndice II.

O nó *Paraná* está localizado na cidade de Curitiba e não possui servidores instalados nele. Entretanto, os nós *Bahia* e *Rio de Janeiro* possuem servidores instalados, com destaque para o nó *Rio de Janeiro* que além de possuir servidor, está definido como *default* e conseqüentemente recebe requisições de clientes que não pertençam a nenhum bloco de endereço descrito no documento de configuração.

```
<netnodes>
  <node name="Parana" local="Curitiba" latitude="-25.42777778"
        longitude="-49.27305556" default="no" hasServer="no"/>
  <node name="Bahia" local="Salvador" latitude="-12.97111111"
        longitude="-38.51083333" default="no" hasServer="yes"/>
  <node name="Rio de Janeiro" local="Rio de Janeiro" latitude=-24.33377778"
        longitude="-45.2075" default="yes" hasServer="yes"/>
</netnodes>
```

Figura 16 – Descrição do elemento <netnodes>.

¹¹ *Coordenadas geográficas* – Sistema de coordenadas medida em graus através da latitude (paralelos) e longitude (meridianos) de um ponto geográfico.

4.1.2 – Blocks

O elemento `<blocks>` descreve os blocos de endereços alocados aos nós da rede de distribuição. Cada nó é descrito pelo elemento `<range>`, cujo atributo `name` identifica o nome do nó. Por sua vez, os blocos de endereços de um determinado nó são identificados por diversos elementos `<block>`. A Figura 17 ilustra a representação do elemento `Blocks`.

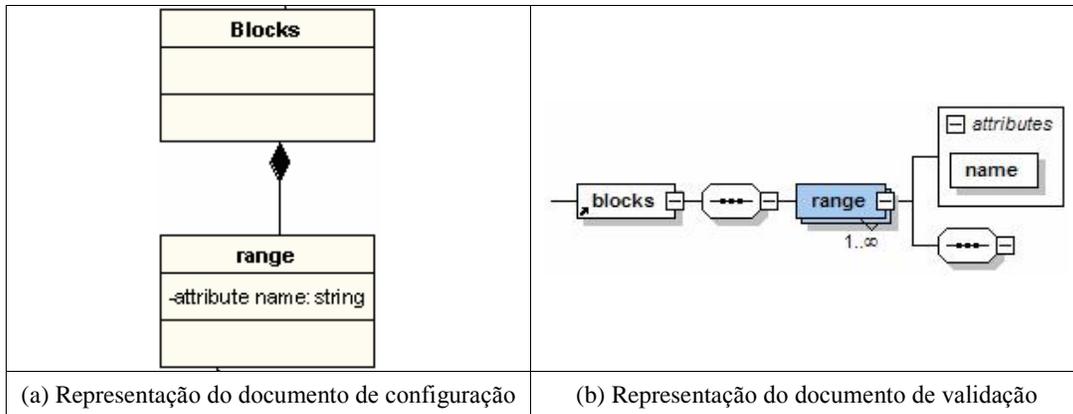


Figura 17 – Representação do elemento `<blocks>`

O elemento `<block>` possui atributos que descrevem o endereço base do bloco e sua respectiva máscara de rede. Conjuntamente, o endereço base e a máscara definem um bloco de endereço, ou seja, um conjunto contíguo de endereços IP. São representados no documento de configuração pelos atributos `address` e `mask` respectivamente. Essas informações são primordiais para o funcionamento do sistema, pois, através desses atributos, o coordenador descobre a localização do cliente. A Figura 18 ilustra o elemento `<block>`, bem como seus atributos.

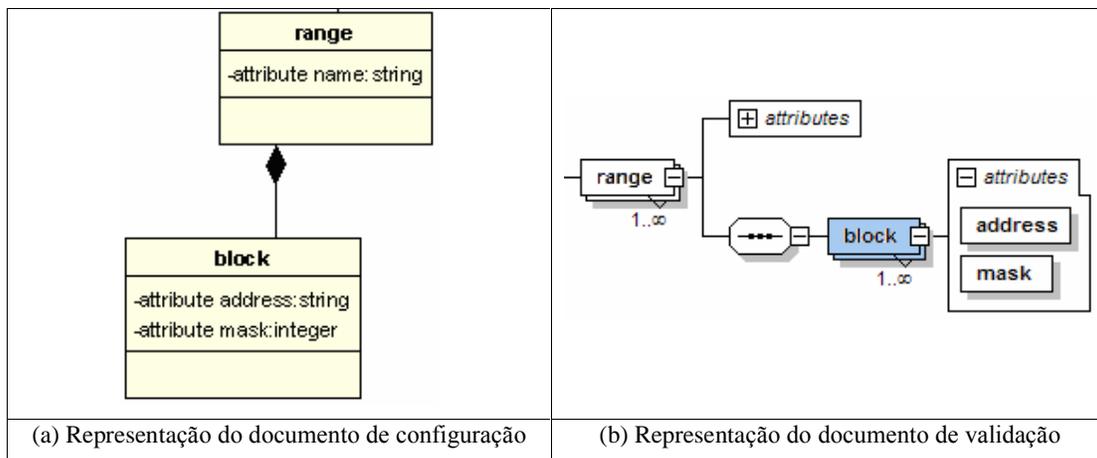


Figura 18 – Representação do elemento `<block>`

- ✓ **address:** O atributo *address* corresponde ao endereço base de bloco alocado para cada *netnode*. Possui o formato de endereçamento IP, são representados por caracteres numéricos inteiros, possui quatro octetos com valores que podem variar de “0” a “255”, sendo estes separados por ponto (.);
- ✓ **mask:** Este atributo representa a máscara de rede correspondente ao endereço do bloco especificado no atributo *address*.

Para uma melhor compreensão de como o elemento `<blocks>` está descrito, a Figura 19 ilustra um pequeno exemplo que descreve o nó *Paraná*, juntamente com os endereços base dos seus blocos e suas respectivas máscaras de rede.

```

<blocks>
  <range name="parana">
    <block address="200.129.173.0" mask="24" />
    <block address="220.132.0.0" mask="20" />
    <block address="150.143.128.0" mask="18" />
  </range>
</blocks>

```

Figura 19 - Descrição do elemento `<blocks>`

4.1.3 – Connections

O elemento `<connections>` representa as conexões físicas existentes entre os nós da rede de distribuição. O elemento `<source>` identifica as conexões originadas a partir de um determinado nó, especificado pelo atributo *name*. O elemento `<target>` descreve o nó de destino da conexão, identificado pelo atributo *name*. Os atributos *rate* e *unit* descrevem a taxa de transmissão e sua respectiva unidade de medida (*bps*, *Kbps*, *Mbps* e *Gbps*), respectivamente. A Figura 20 ilustra o elemento `<connections>` descrito no documento de configuração bem como no documento de validação.

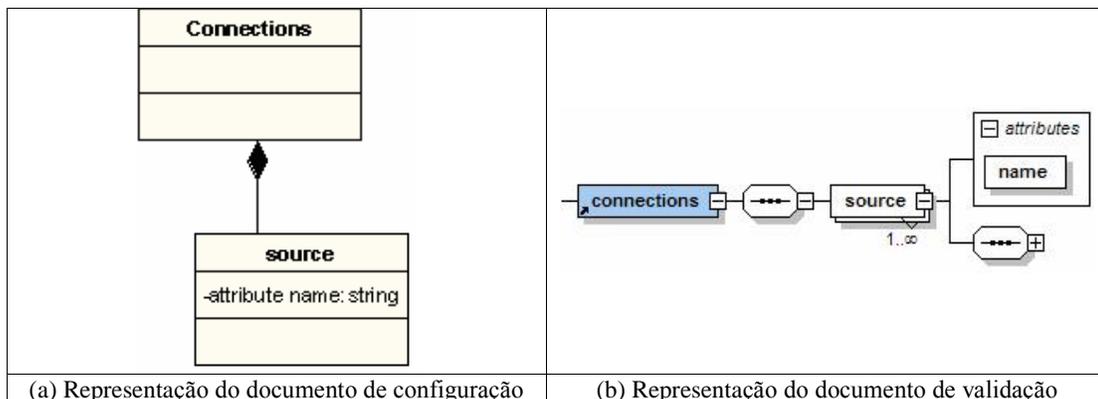


Figura 20 – Estrutura do elemento `<connections>`.

A Figura 21 ilustra o elemento `<source>` descrito no documento de configuração bem como no documento de validação.

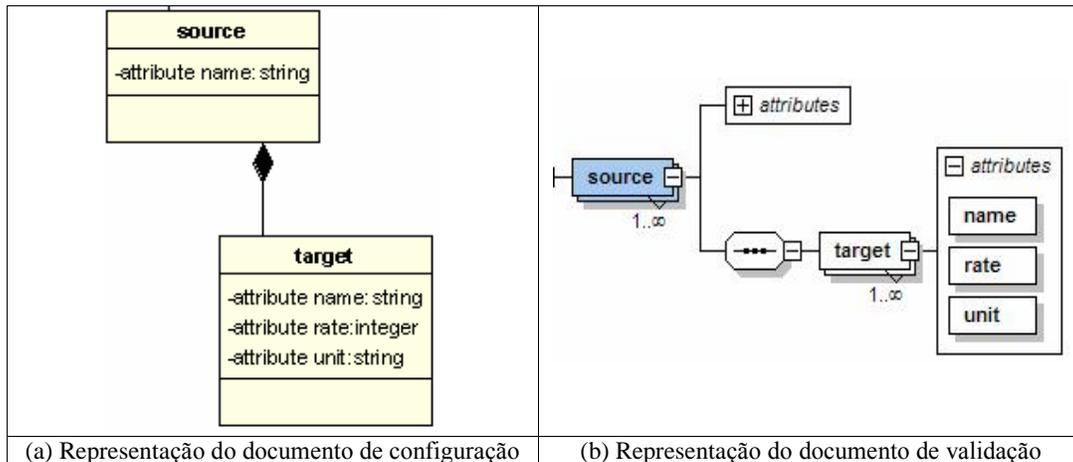


Figura 21 – Representação do elemento `<source>`.

Para uma melhor compreensão de como o elemento `<source>` está descrito no documento de configuração, a Figura 22 ilustra o nó *Parana*, bem como suas respectivas conexões como os nós: *Sao Paulo* (155 Mbps), *Rio de Janeiro* (155 Mbps) e *Santa Catarina* (80 Mbps).

```
<connections>
  <source name="Parana">
    <target name="Sao Paulo" rate="155" unit="Mbps" />
    <target name="Rio de Janeiro" rate="155" unit="Mbps" />
    <target name="Santa Catarina" rate="80" unit="Mbps" />
  </source>
</connections>
```

Figura 22 – Descrição do elemento `<connections>`

4.1.4 – Servers

O elemento `<servers>` representa os servidores disponíveis no serviço de distribuição de vídeo. É composto pelo elemento `<pool>` que descreve o nome do nó que possui servidor instalado, bem como o estado atual do mesmo. Isso é descrito através dos atributos `name` e `status`, respectivamente.

A Figura 23 ilustra o elemento `<servers>`, seu elemento filho `<pool>` juntamente com seus atributos.

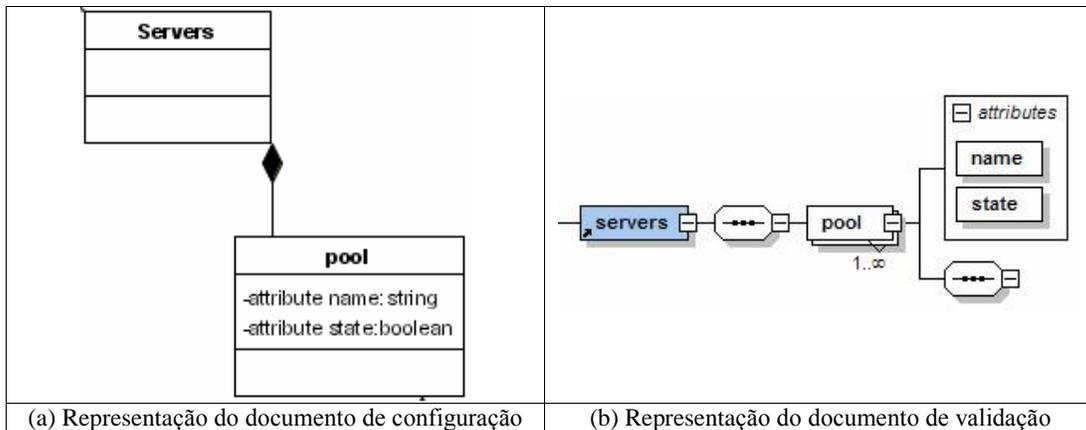


Figura 23 – Representação do elemento <servers>

O elemento *<pool>* contém um elemento denominado *<server>*, cujo atributo *ip* identifica o endereço IP do servidor. A Figura 24 ilustra o elemento *<pool>*.

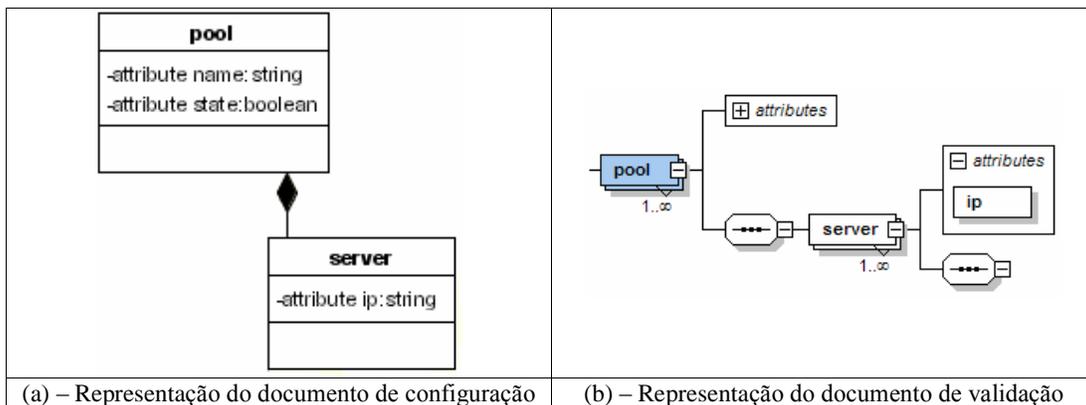


Figura 24 – Representação do elemento <pool>.

O elemento *<service>* possui dois atributos *name* e *port* que definem o tipo de serviço que um servidor disponibiliza aos clientes – ao vivo (*D-live*) ou sob demanda (*D-VoD*) – bem como a porta que o servidor recebe requisições. A Figura 25 ilustra o elemento *<service>* juntamente com seus atributos *name* e *port*.

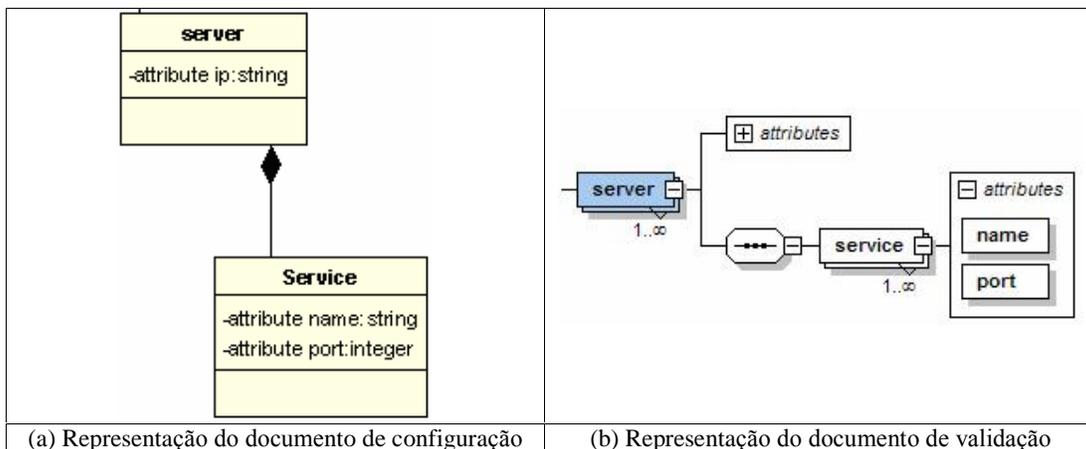


Figura 25 – Representação do elemento <service>.

A Figura 26 ilustra um exemplo de um servidor representado no documento de configuração.

```

<pool name="Rio de Janeiro">
  <server ip="200.159.254.151">
    <service name="dvod" port="9998" />
    <service name="dlive" port="6500" />
  </server>
</pool>

```

Figura 26 – Descrição do elemento <servers>

Como descrito na Figura 26, o nó *Rio de Janeiro* possui um servidor com endereço IP *200.159.254.151*. Este servidor disponibiliza dois tipos de serviços aos clientes: sob-demanda, através da porta 9998 e ao vivo, através da porta 6500.

Para uma visão geral dos elementos contidos no documento de configuração, a Figura 27 ilustra um diagrama completo dos elementos contidos nesse documento, bem como seus atributos e tipos definidos.

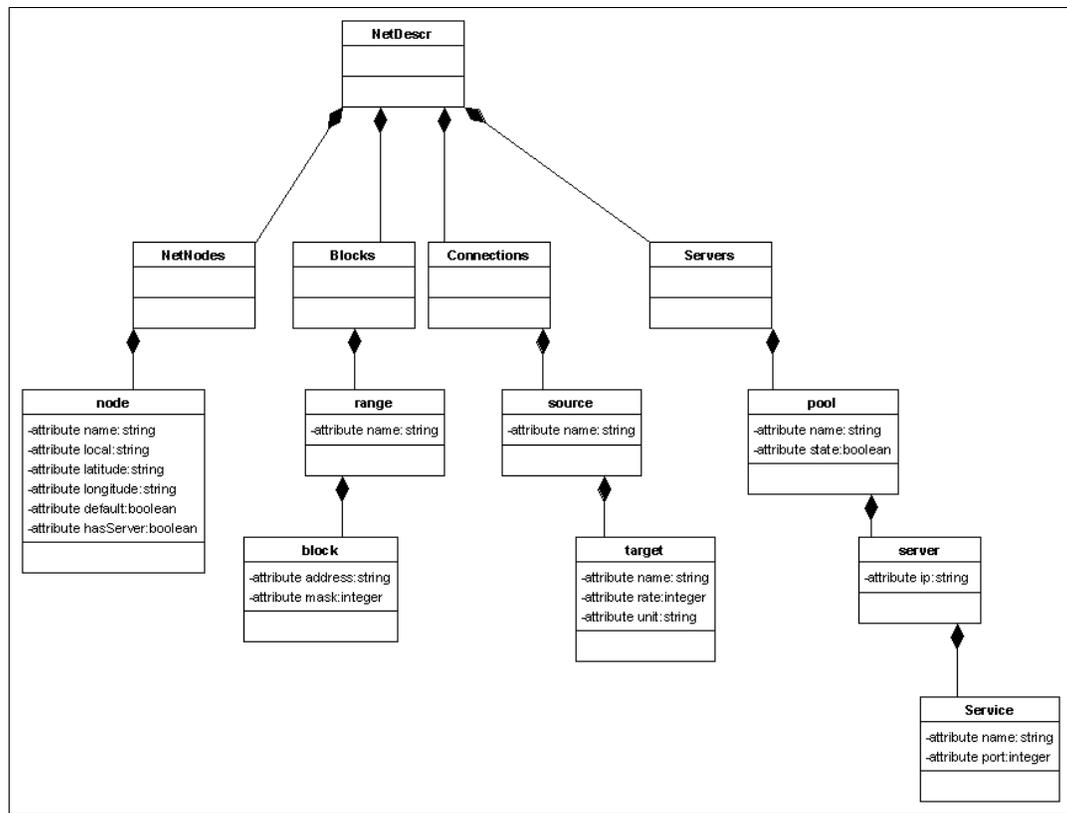


Figura 27 – Diagrama dos elementos – Documento de configuração

4.2 – Considerações Finais

Esse Capítulo apresentou a topologia da rede de distribuição de vídeo digital, descrevendo sua especificação em um documento de configuração que descreve a rede de distribuição através de elementos e atributos e que é caracterizado por informações essenciais para o sucesso das estratégias utilizadas para distribuição de vídeo.

Capítulo 5

Configuração e Gerenciamento do Serviço de Distribuição de Vídeo Digital

Este capítulo descreve a Aplicação de Configuração denominada Configurator. Concebida e desenvolvida para manipular as informações contidas no documento de configuração do serviço de distribuição de vídeo digital. Para tanto, é apresentado um modelo UML (diagramas de casos de uso), que descreve as funcionalidades dessa aplicação. Posteriormente, será apresentada a aplicação propriamente dita.

5.1 – Diagrama de Casos de Uso UML

Esta seção apresenta os diagramas de casos de uso, descrevendo as funcionalidades e relacionamentos existentes entre eles. Um diagrama de caso de uso UML descreve as operações que o sistema disponibiliza ao administrador do sistema, ajudando assim a formalizar as funções do sistema.

O diagrama ilustrado pela Figura 28 descreve as funcionalidades do Configurator – desenvolvida para a configuração e gerenciamento das informações contidas no documento de configuração do serviço de distribuição de vídeo.

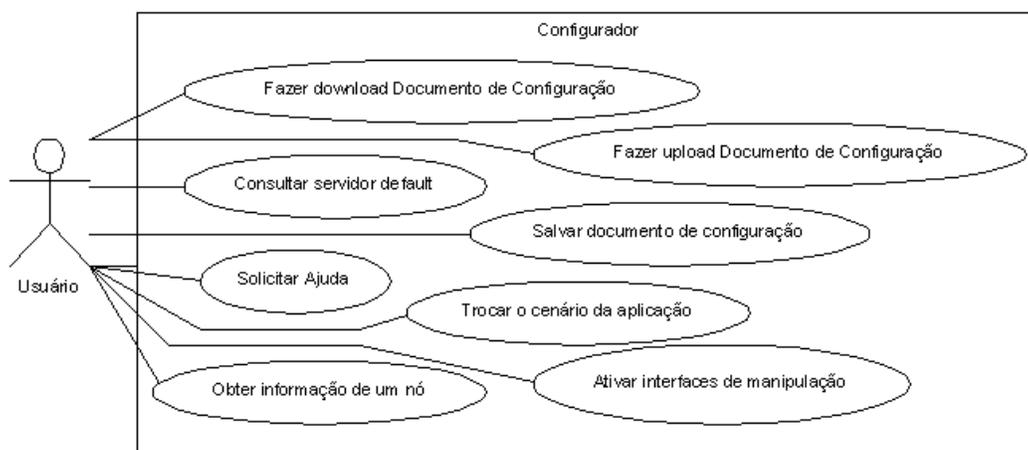


Figura 28 – Diagrama UML do Configurator

- *Fazer Download Documento de Configuração* – Qualquer manipulação dos dados contidos no documento de configuração deve ser realizada localmente na máquina do administrador. Essa opção disponibiliza uma

cópia do documento de configuração para que o administrador possa manipular as informações contidas no mesmo;

- *Fazer Upload Documento de Configuração* – Esta opção disponibiliza o documento de configuração ao coordenador, que por sua vez atualiza as novas informações no sistema de distribuição;
- *Consultar servidor default* – Além de adoção de cores distintas na representação dos nós, servidores e servidor padrão, foi definido uma alternativa manual que permite ao administrador obter informações referentes ao servidor definido como padrão (*default*) no serviço de distribuição;
- *Obter informações de um nó* – Disponibiliza informações gerais de um nó contido no documento de configuração. Essas informações são: nome do nó, localização, bem como sua posição geográfica. Informa a existência de um servidor e serviços disponíveis, dentre outros;
- *Solicitar Ajuda* – Permite ao administrador realizar consultas referentes aos elementos contidos no documento de configuração;
- *Ativar interfaces de manipulação* – Refere-se à configuração do serviço propriamente dita, através da ativação das interfaces definidas para a manipulação dos dados contidos no documento de configuração. Para melhor compreensão, iremos detalhar as funcionalidades existentes nessa opção através dos módulos *Netnodes*, *Connections*, *Blocks*, e *Servers*.

O módulo *NetNodes* descreve os nós existentes na rede de distribuição. A Figura 29 ilustra esse módulo, bem como suas funcionalidades.

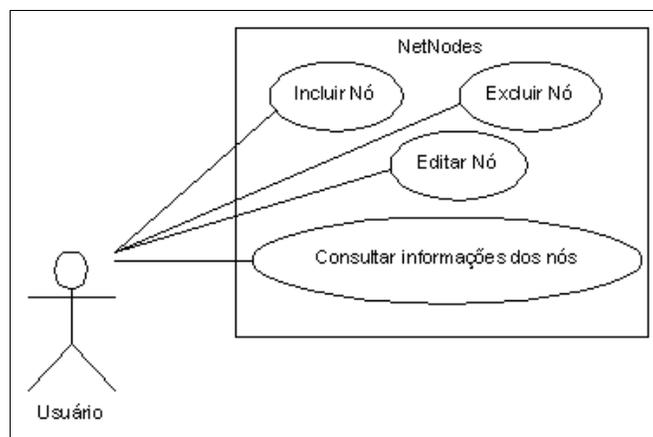


Figura 29 – Diagrama UML da classe *NetNodes*

As operações que o administrador dispõe para a manipulação dos nós existentes no documento de configuração através do elemento <netnodes> são:

- *Incluir nó* – Essa opção permite ao administrador adicionar um nó com todas suas características como: nome, localização e posição geográfica no documento de configuração;
- *Excluir nó* – Permite a exclusão de um nó e suas características. Deve ser ressaltado que a exclusão de um nó só é possível se este não possuir nenhuma rede (endereço base) alocada a ele;
- *Editar nó* – Esta opção permite ao administrador editar informações como nome, localização e posição geográfica (latitude e longitude), referentes aos nós contidos no documento de configuração;
- *Consultar nó* – Essa opção permite ao administrador obter informações referentes aos nós existentes no documento de configuração.

O módulo *Connections* descreve as conexões existentes entre nós e servidores. A Figura 30 ilustra esse módulo, bem como suas funcionalidades.

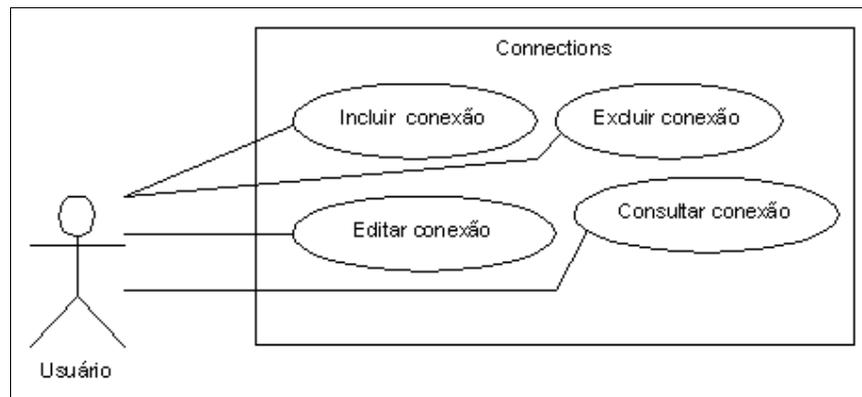


Figura 30 – Diagrama UML da classe *Connections*

As operações que o administrador dispõe para a manipulação dos nós existentes no documento de configuração através do elemento <connections> são:

- *Incluir conexão* – Essa opção permite a inclusão de uma conexão e seus respectivos atributos (origem, destino, taxa e unidade) entre dois nós existentes no documento de configuração;
- *Excluir conexão* – Essa opção exclui uma conexão (origem/destino) e seus respectivos atributos do documento de configuração;
- *Consultar conexão* – Permite ao administrador consultar a existência de uma conexão entre dois nós;

- *Editar conexão* – Permite editar as informações como taxa (velocidade da conexão) e a unidade da mesma (*bps, Mbps, Gbps e Tbps*), referentes à conexão entre dois nós.

O módulo *Blocks* descreve os blocos de endereços IP alocados a um nó. A Figura 31 descreve esse módulo, bem como suas funcionalidades.

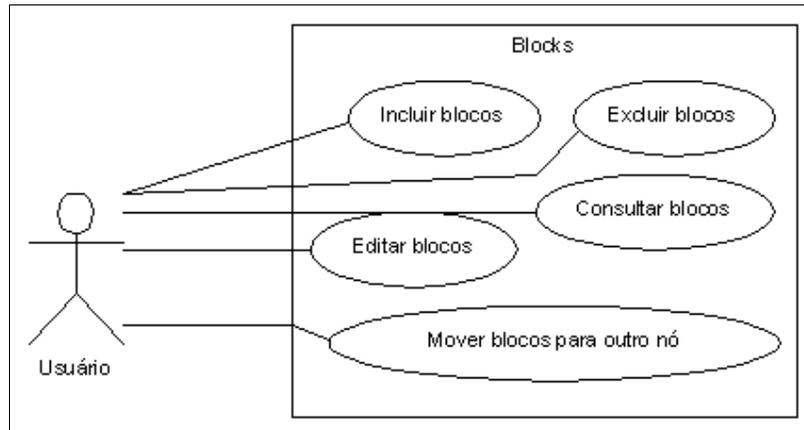


Figura 31 – Diagrama UML da classe *Blocks*

As operações que o administrador dispõe para a manipulação dos nós existentes no documento de configuração através do elemento *<blocks>* são:

- *Incluir blocos* – Permite a alocação de um conjunto contíguo de endereços IP a um determinado nó;
- *Excluir blocos* – Permite a exclusão de um conjunto contíguo de endereços IP alocados a um nó;
- *Consultar blocos* – Obtém informações referentes aos blocos de endereços alocados a cada nó disponível no documento de distribuição;
- *Mover blocos para outro nó* – Essa funcionalidade permite ao administrador mover um ou vários blocos de endereço IP de um nó para outro existente no documento de configuração.

O módulo *Servers* descreve os servidores, bem como os serviços existentes no serviço. A Figura 32 descreve esse módulo, bem como suas funcionalidades.

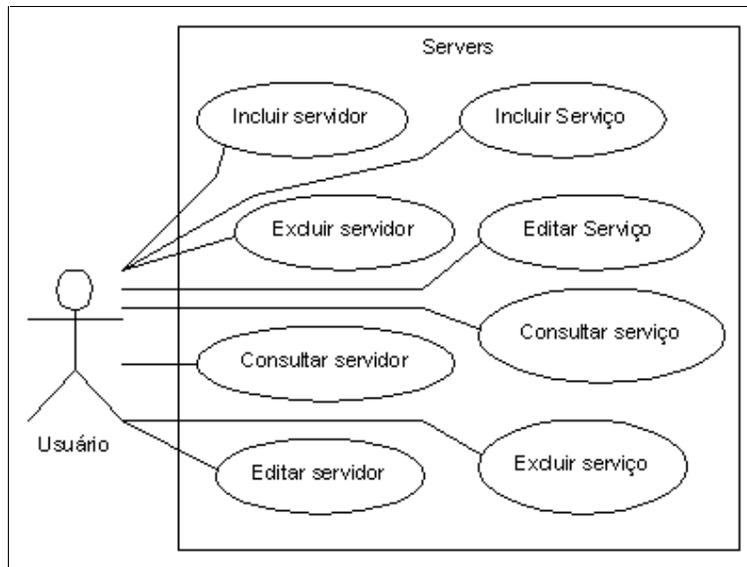


Figura 32 – Diagrama UML da classe *Servers*

As operações que o administrador dispõe para a manipulação dos nós existentes no documento de configuração através do elemento `<servers>` são:

- *Incluir servidor* – Permite ao administrador incluir um servidor com todas as características como: nome do nó, endereço IP do servidor, e seu *status* (*online/offline*);
- *Excluir servidor* – Permite a exclusão de um servidor e todos atributos relacionados a ele, como por exemplo, os serviços disponíveis nesse servidor;
- *Consultar servidor* – O administrador pode obter informações referentes aos servidores instalados nos nós disponíveis no documento de configuração;
- *Editar servidor* – Permite ao administrador editar informações referentes aos servidores, como por exemplo, endereço IP e *status* do mesmo;
- *Incluir serviço* – Um servidor deve disponibilizar serviços de distribuição aos clientes (ao vivo ou sob demanda). Essa opção inclui um novo serviço em um servidor pré-definido no documento de configuração;
- *Excluir serviço* – Permite a exclusão de um serviço associado a um servidor;

- *Editar serviço* – Através dessa opção é possível editar informações referentes aos serviços associados aos servidores, como por exemplo, nome do serviço e porta que o mesmo espera por requisições;
- *Consultar serviço* – O administrador pode obter informações referentes aos serviços disponíveis em um servidor.

5.2 – Visão Geral do Configurador

Essa seção apresenta o protótipo da aplicação desenvolvida para configuração e gerenciamento das informações contidas no documento de configuração do serviço de distribuição de vídeo.

O configurador foi desenvolvido usando a linguagem Java, favorecendo a sua portabilidade com diversas plataformas. Possui uma interface principal que permite a configuração e o gerenciamento das informações contidas no documento de configuração, proporcionando um conjunto consistente de componentes intuitivos de interface com o usuário.

O configurador ilustra os nós, conexões e servidores (ativos e inativos) que compõem a rede de distribuição. Vale ressaltar que os nós são posicionados no configurador através do mapeamento aproximado de suas respectivas coordenadas geográficas (latitude e longitude). A partir desta interface principal é possível ativar outras interfaces que permitem a manipulação das informações contidas no documento de configuração através de operações de inserção, remoção, alteração e consulta, proporcionando um conjunto consistente de componentes de interface com o usuário.

Para manipular as informações contidas no documento de configuração, o administrador pode ativar essas funcionalidades clicando com o botão direito do mouse sobre o nó, servidor ou conexão a ser manipulada. Utilizando essas funcionalidades, o administrador pode incluir, remover, modificar ou consultar informações do documento de configuração. A seguir serão descritas algumas funcionalidades providas pelo configurador. A interface principal pode ser visualizada na Figura 33.



Figura 33 – Aplicação gráfica principal

Os pontos representam os nós, bem como os servidores existentes na rede de distribuição. Os nós *Rio Grande do Sul*, *Santa Catarina*, *São Paulo*, *Rio de Janeiro*, *Fio Cruz*, *Distrito Federal*, *Pernambuco*, *Paraíba*, *Rio Grande do Norte* e *Ceará* representam os servidores em atividade no sistema de distribuição de vídeo. Por sua vez, descrevem a origem de uma conexão. Como já mencionado anteriormente, o servidor instalado no nó *Rio de Janeiro* é definido como servidor padrão do sistema de distribuição, por isso sua representação em cor diferenciada.

Na próxima seção apresentaremos o detalhamento da aplicação-piloto desenvolvida no contexto da RNP.

5.2.1 – Mapeamento dos nós na tela

Os pontos ilustrados pela Figura 33 representam os nós contidos no documento de configuração. Estes nós são mapeados de acordo com uma aproximação real de sua coordenada geográfica (*Latitude e Longitude*). Através da coordenada geográfica de um nó, é realizado um mapeamento para pontos de tela e posteriormente, são ilustrados no configurador.

Para que os nós sejam representados no configurador, é realizada uma conversão de coordenada geográfica expressa em graus, minutos, segundos ($46^{\circ}51'39''$) para graus decimais (-46.86083333). Os valores que levam o sinal negativo são os pontos localizados a Oeste (*West*) e Sul (*South*). Vale ressaltar que os valores em graus

decimais correspondem à coordenada geográfica convertidas para segundos. O resultado dessa conversão é submetido a novos cálculos matemáticos, para então obter os pontos de tela. Assim é realizado o mapeamento, com base nas coordenadas do cenário que representa a imagem utilizada pelo configurador.

5.2.2 – *Download e UpLoad do Documento de Configuração*

A manipulação dos dados contidos no documento de configuração deve ser realizada localmente, ou seja, se faz necessário que este esteja armazenado na máquina do administrador. A réplica desse documento é solicitada pelo administrador ao coordenador. A Figura 34 ilustra as opções de *download* e *upload* do documento de configuração.

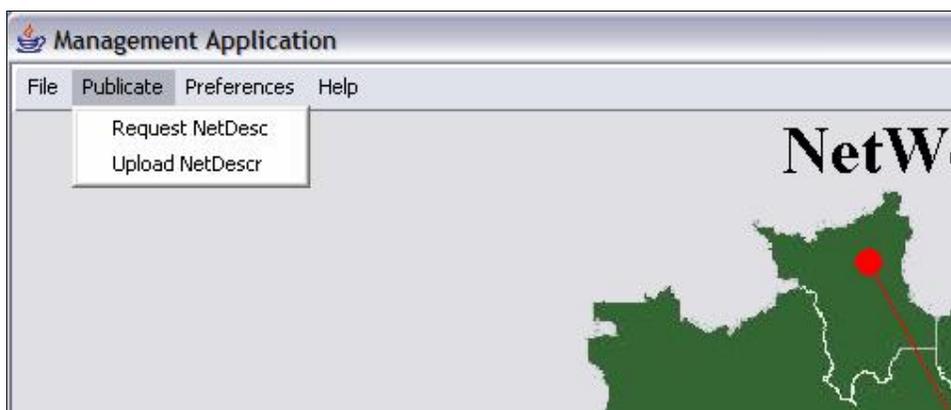


Figura 34 – *Download e UpLoad do Documento de Configuração*

Após selecionar a opção *Publish/Request NetDesc*, o administrador receberá o documento de configuração localmente, estando apto à manipulação dos dados (*download*). Para que as novas informações sejam atualizadas no sistema, o administrador deverá enviar o documento de configuração para o coordenador (*upload*). Dessa forma, as novas informações são atualizadas no grafo de conectividade disponível no sistema de distribuição. A opção *Publish/Upload NetDesc*, antes de enviar o documento ao coordenador, verifica se as novas informações contidas nele encontram-se de acordo com as regras impostas pelo documento de validação. Estando os dados válidos, o documento é publicado pelo coordenador. Caso contrário, uma mensagem de erro é apresentada ao administrador, descrevendo qual é a informação inválida no documento de configuração.

5.2.3 – Obter informação dos nós e estado operacional dos servidores

O configurador provê mecanismos que permitem ao administrador do serviço de distribuição de vídeo obter informações sobre os nós e identificar o estado operacional dos servidores existentes na rede de distribuição.

As informações resumidas de cada nó podem ser obtidas passando o cursor do mouse sobre o nó que se deseja obter informação. A Figura 35 ilustra as informações básicas como nome do nó, sua localidade, bem como sua coordenada geográfica referentes ao nó *Rio de Janeiro*.

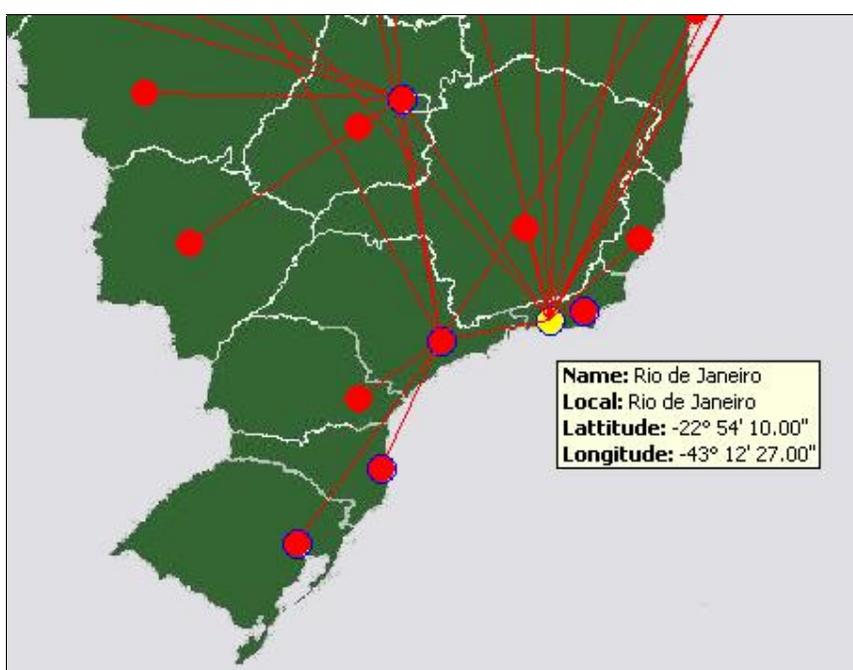


Figura 35 – Obter informações de um nó existente na rede de distribuição.

O estado ativo/inativo dos servidores, bem como a representação do servidor *default* já é explicitamente representado na interface principal adotando cores distintas.

5.2.4 – Funcionalidade de zoom sobre o cenário

A funcionalidade de zoom permite ao administrador aumentar a imagem utilizada pelo configurador em uma quantidade indefinida de vezes. Essa funcionalidade é de grande utilidade para o administrador, tanto para um melhor posicionamento dos pontos na tela (precisão), como também para uma melhor visualização de uma quantidade relativamente grande de pontos em uma mesma região. Vale ressaltar que cada clique sobre o botão de aumento de zoom, representa um aumento de 10% em

relação ao tamanho atual. O mesmo vale quando o administrador clicar no botão de redução do zoom, a cada clique o cenário será diminuído de 10%. A Figura 36 ilustra a funcionalidade de zoom, a imagem aumentada e o reposicionamento dos nós.

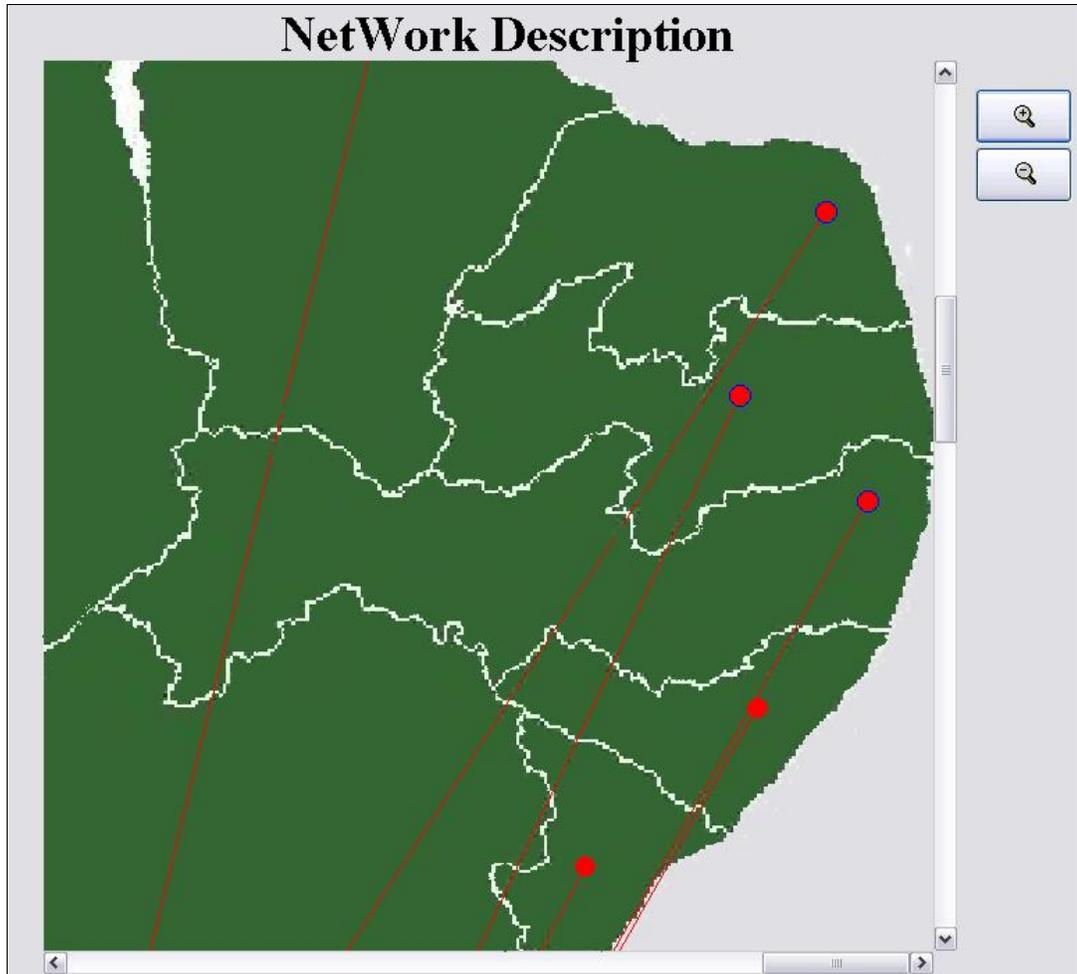


Figura 36 – Funcionalidade de zoom

5.2.5 – Configuração dos Nós

Os nós são referências para qualquer manipulação de informação no documento de configuração. Para inserir novos blocos de endereços IP, servidores ou conexões no documento de configuração, um nó existente deve ser selecionado para posteriormente ser editado.

Para inserir um novo nó no documento de configuração, o administrador deve ativar a interface de inserção de um nó. Para tal, deve clicar o botão direito do mouse na posição que deseja inserir o nó e selecionar a opção *NetNodes/Add*. A Figura 37 ilustra o procedimento que ativa a interface de inserção de nó.

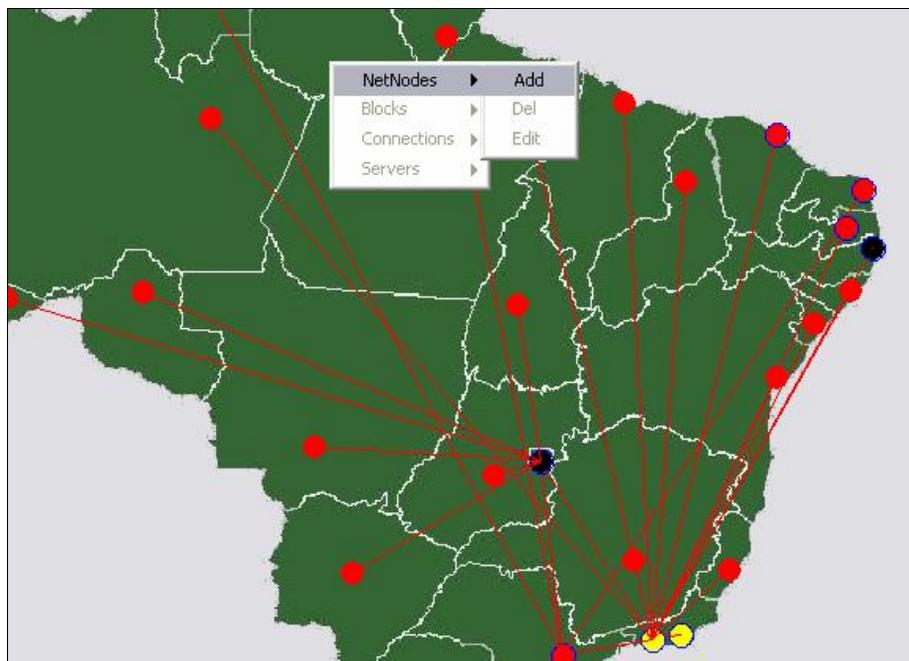


Figura 37 – Inserção de um nó

Após selecionar a opção de inserção de um nó, a interface ilustrada na Figura 38 é apresentada ao administrador. O campo *Name* identifica o nome do nó, que geralmente representa um ponto de presença (POP) ou instituição que faz uso do serviço de distribuição de vídeo. O campo *Local* identifica a localização física do nó, geralmente indicada pelo nome da cidade onde o mesmo se encontra. O valor aproximado da coordenada geográfica do nó (latitude/longitude) é calculado automaticamente pela aplicação a partir das coordenadas do ponto selecionado, podendo ser atualizado pelo administrador.

A dialog box titled "NetNode" with a close button in the top right corner. The dialog contains a form with the following fields:

NetNode	
Name	Local
<input type="text" value="Lavid"/>	<input type="text" value="Joao Pessoa"/>
Latitude	Longitude
<input type="text" value="-35° 12 '19.67"/>	<input type="text" value="-06° 55 '33.32"/>

At the bottom of the dialog are two buttons: "Ok" and "Cancel".

Figura 38 – Interface de inserção de um novo nó

A inserção de campos em branco, valores inválidos ou inadequados não são aceitos por essa interface, conforme descrito anteriormente no documento de validação.

Analogamente ao procedimento de inserção de um nó no documento de configuração, o administrador também dispõe da funcionalidade de *Edição* de um nó. Após clicar com o botão direito do mouse sobre o nó que se deseja editar, a interface ilustrada pela Figura 38 será disponibilizada ao administrador com as informações referentes ao nó selecionado, podendo este ser modificado com informações válidas, restritas pelo documento de validação.

Para remover um nó do documento de configuração o administrador deve clicar com o botão direito do mouse sobre o nó que se deseja remover. Utilizando esses procedimentos, o administrador ativa a interface de remoção ilustrada pela Figura 39.

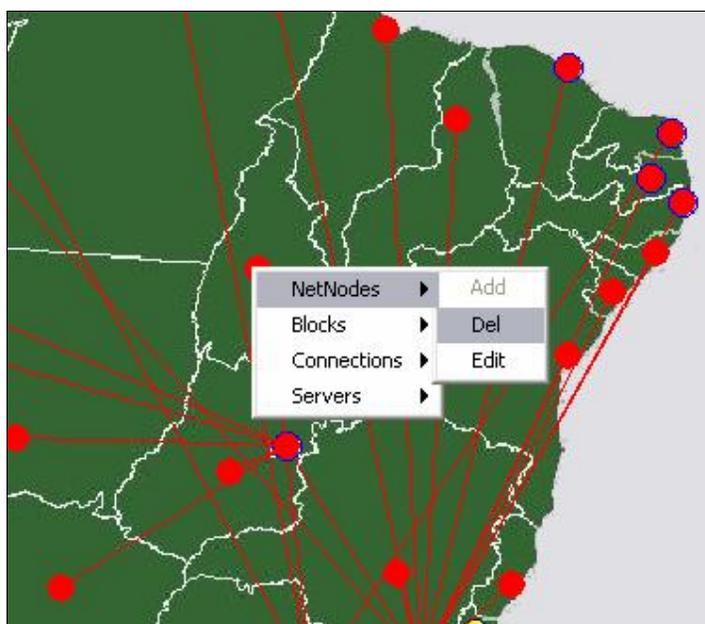


Figura 39 – Chamada da interface de remoção de um nó

Após selecionar o nó que será removido do documento de configuração (*Piauí*), uma mensagem de confirmação é apresentada ao administrador para que este possa efetuar a operação de remoção. A Figura 40 ilustra a interface contendo a confirmação da operação.



Figura 40 – Confirmação de remoção de um nó

Vale ressaltar que um nó contendo blocos de endereços IP não pode ser totalmente removido do documento de configuração sem antes ter seus blocos removidos ou alocados a outro nó existente nesse documento. A Figura 41 ilustra a interface que permite ao administrador mover os blocos de endereço IP de um nó para outro existente no documento de configuração.

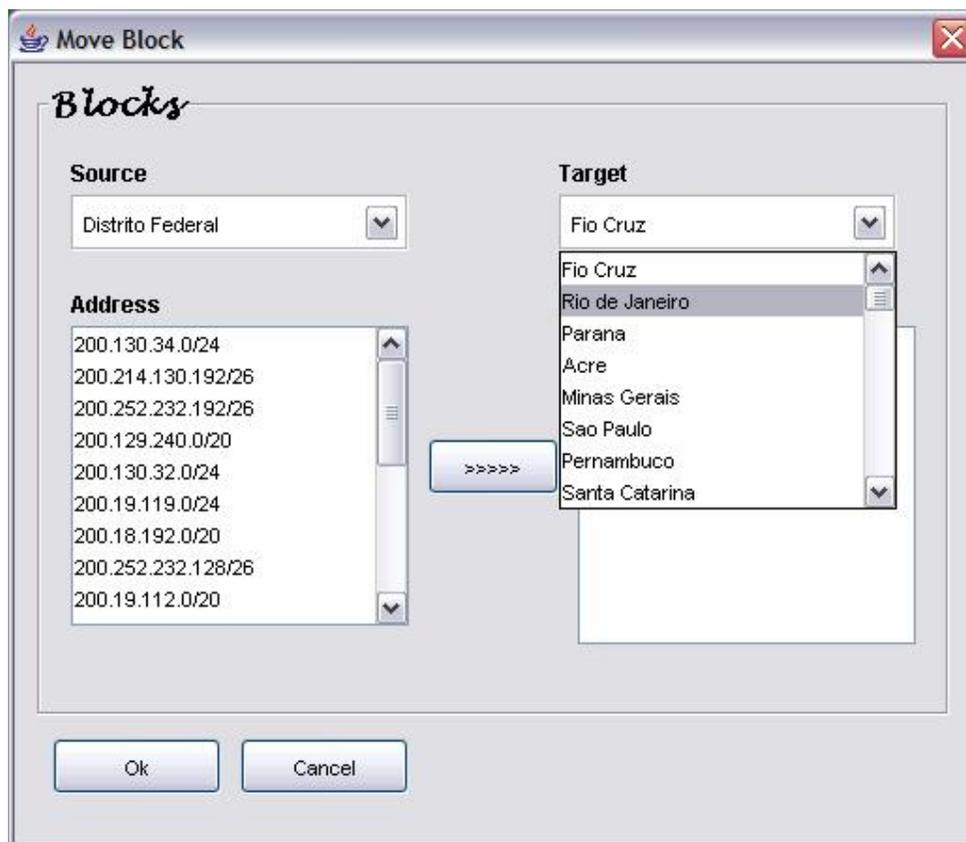


Figura 41 – Movendo blocos para outro nó

No exemplo ilustrado pela Figura 41, o campo *Source*, representa o nó de origem dos blocos de endereços IP, preenchido pelo nó *Distrito Federal*. No campo *Target* é ilustrado a lista de nós disponíveis no documento de configuração aptos a receber os novos blocos de endereços. No exemplo da Figura 41, o nó *Rio de Janeiro* está sendo selecionado para receber os novos blocos de endereços IP.

Além das informações visuais que a aplicação disponibiliza ao administrador, este pode ativar a interface apenas para consulta de informações dos nós existentes no documento de configuração. Essa interface de consulta contém todas as informações referentes aos nós na guia *Nodes*, blocos de endereços na guia *Blocks*, conexões existentes na guia *Connections*, servidores e serviços associados aos nós na guia *Servers*. Para ativá-la, o administrador deve utilizar a funcionalidade de clique duplo do mouse sobre o nó que se deseja consultar. A Figura 42 ilustra a interface ativada no procedimento para consulta de um nó contido no documento de configuração.

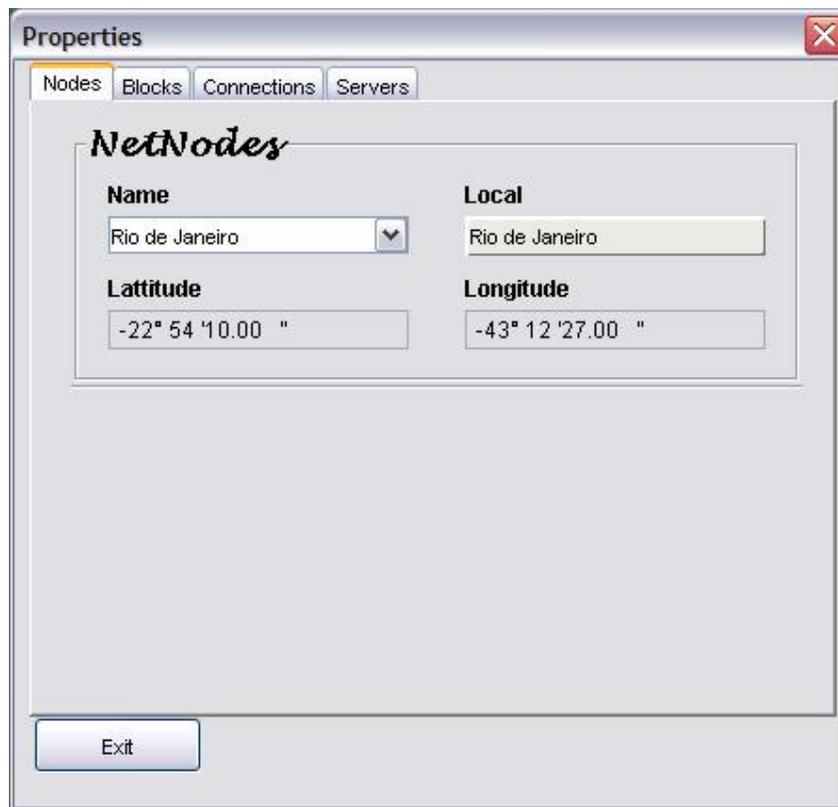


Figura 42 – Consulta de um nó na rede de distribuição

A guia *Nodes* ilustra as informações referentes ao nó *Rio de Janeiro*, sua localidade e suas respectivas coordenadas geográficas.

Os blocos de endereços IP alocados ao nó *Rio de Janeiro* pode ser visualizado na Figura 43.

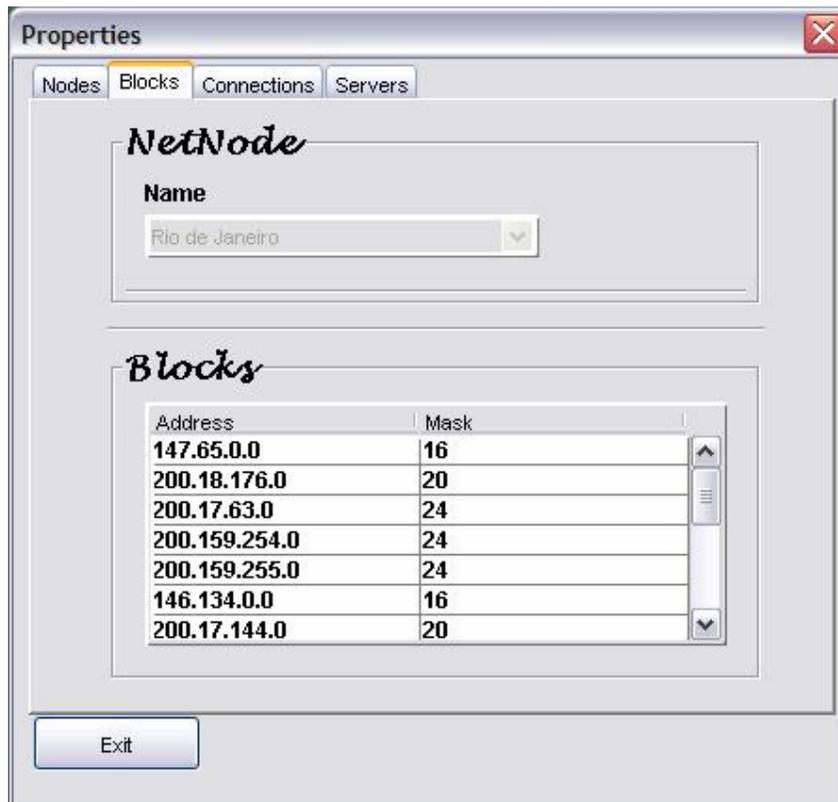


Figura 43 – Consulta a blocos alocados ao nó *Rio de Janeiro*

A Figura 44 ilustra os nós que possuem conexão com o nó *Rio de Janeiro*. O campo *Source* descreve o nó que origina a conexão (*Rio de Janeiro*). No campo *Target* é descrito a lista de nós que possuem conexão com o nó *Rio de Janeiro*, enquanto que os campos *Rate* e *Unit* descrevem a velocidade da conexão (622) e a unidade da mesma (*Mbps*), respectivamente. De acordo com o nó selecionado no campo *Target*, os campos *Rate* e *Unit* são automaticamente atualizados com as respectivas informações da conexão em questão.



Figura 44 – Consulta as conexões existentes com o nó *Rio de Janeiro*

A Figura 45 ilustra os servidores, bem como os serviços disponíveis no nó *Rio de Janeiro*. Como pode ser visualizado, o campo *Node* descreve o nome do nó que possui o servidor instalado nele. No campo *IP Address* é descrito a lista de servidores que este nó possui. No exemplo da Figura 45, o nó *Rio de Janeiro* possui apenas um servidor, tendo associado a ele dois serviços, *dvod* e *dlive*. O campo *Port* informa a porta em que este serviço está rodando, enquanto que o campo *Status* informa o estado atual deste servidor bem como se ele está definido como servidor *default* ou não no documento de configuração.



Figura 45 – Consulta a servidores associados ao nó *Rio de Janeiro*

5.2.6 – Configuração de blocos de endereço IP

Novos blocos de endereço IP podem ser adicionados aos nós existentes no documento de configuração. Para isso, a aplicação verifica se o novo bloco já se encontra em uso por outro nó. Caso isso ocorra, uma mensagem de erro é disponibilizada ao administrador e o bloco não é adicionado no documento de configuração. Para validar um endereço IP de um bloco, o configurador realiza a operação lógica *or* entre o endereço IP do bloco e a máscara desse bloco. O resultado dessa operação lógica deve ser igual à própria máscara. Se o resultado da operação é igual a máscara desse bloco, então o endereço é válido e pode ser adicionado ao documento de configuração.

Utilizando os mesmos procedimentos ilustrados pela Figura 37, o administrador pode adicionar novos blocos de endereços aos nós disponíveis no documento de configuração. Para ativar a interface de inserção, bem como de edição de um bloco de endereço, o administrador deve clicar com o botão direito do mouse sobre o nó desejado. A Figura 46 ilustra a interface ativada no procedimento para inserção de novos blocos no documento de configuração.

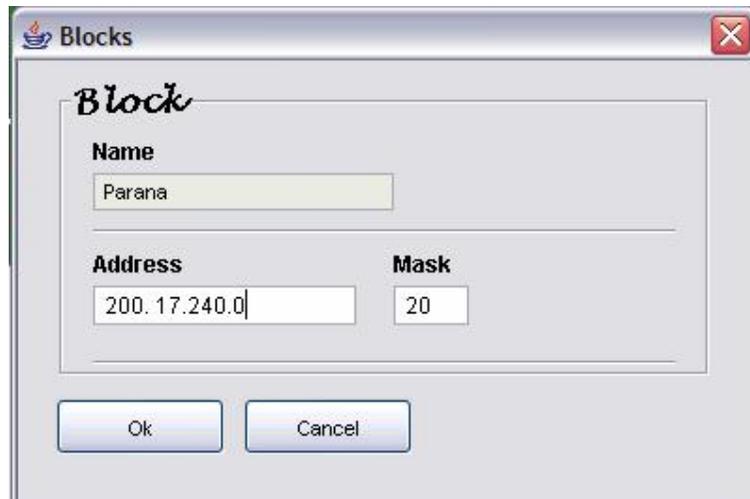


Figura 46 – Inserção de um novo bloco de endereço IP

O exemplo ilustrado pela Figura 46 ilustra a interface que permite a adição do bloco de endereço *200.17.240.0*, máscara de rede *20*, ao nó *Paraná* (pré-selecionado), selecionado no campo *Name*. Antes da inserção desses dados, a aplicação de configuração e gerenciamento faz uma busca no documento de configuração, verificando se esse bloco e máscara já estão em uso por outro nó. Se o bloco não estiver em uso, este é adicionado no documento de configuração. Caso contrário, uma mensagem de erro é apresentada ao administrador.

O administrador também pode remover um bloco de endereço IP clicando com o botão direito do mouse sobre o nó que contém o bloco que será removido. Selecionando a opção *Blocks/Del*, é apresentada ao administrador uma lista de blocos alocados ao nó selecionado. Para total remoção do bloco desejado, basta selecioná-lo, e este será automaticamente removido do documento de configuração.

A Figura 47 ilustra o procedimento utilizado para remoção do bloco de endereço IP *200.137.165.0* com máscara *24*, alocado ao nó *Piauí*.

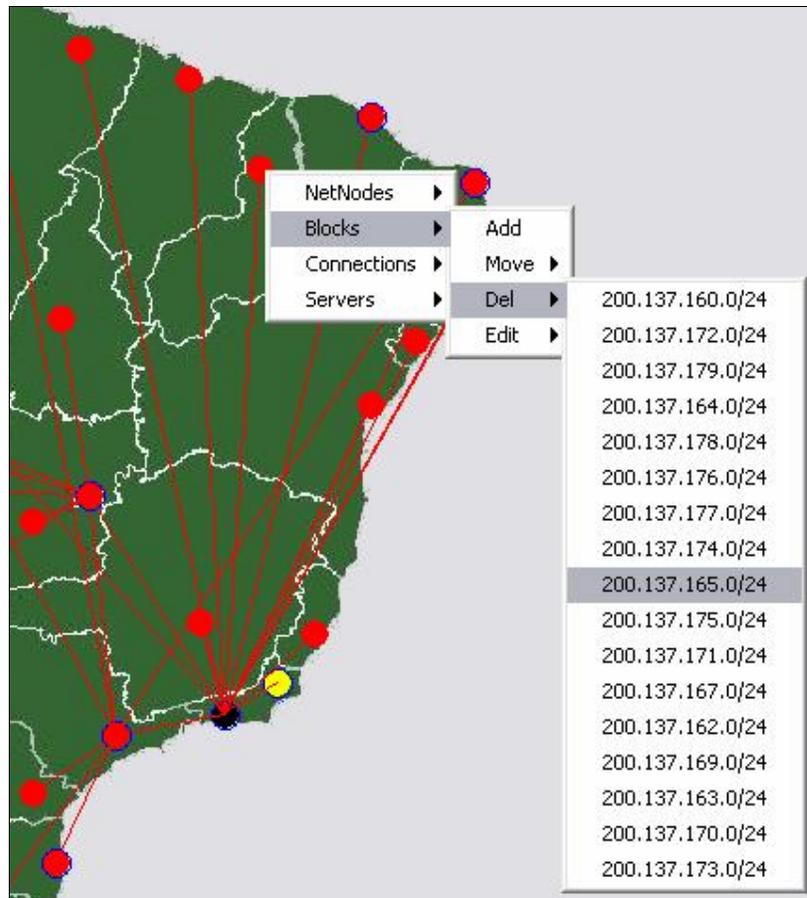


Figura 47 – Remoção de um bloco de endereço IP.

Também foi desenvolvida uma interface que permitisse ao administrador realizar a edição dos blocos de endereço IP, bem como sua respectiva máscara. Para ativar essa interface, o administrador deve clicar com o botão direito do mouse sobre o nó que contém o bloco que será editado, e selecionar a opção *Block/Edit*.

Após selecionar o nó que contém o bloco de endereço que se deseja editar, a Figura 48 ilustra a interface ativada que é utilizada na alteração de um bloco de endereço contido no documento de configuração. Campos em branco e valores inválidos não são permitidos nessa interface. O configurador faz uma busca ao documento de configuração para verificar se o bloco a ser inserido já está em uso por outro nó. Caso não esteja, é incluído no documento de configuração. Caso contrário, uma mensagem de erro é apresentada ao administrador da aplicação. Vale ressaltar a metodologia adotada para validar um endereço IP de um bloco, é o mesmo adotado na inserção de um bloco.

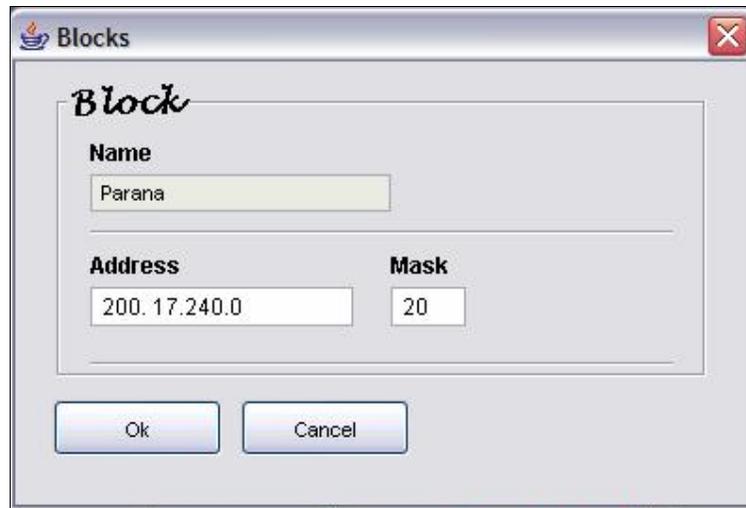


Figura 48 – Substituição de um bloco de endereço IP.

Como ilustrado pela Figura 48, o bloco de endereço *200.17.240.0* pode ser editado ou modificado por qualquer bloco válido e que não esteja em uso por outro nó existente no documento de configuração.

5.2.6.1 – Mover Blocos de endereços IP para outro nó

Para mover blocos de endereços IP para outros nós, o administrador dispõe de duas opções: a primeira é clicar com o botão direito do mouse sobre o nó que contém o bloco de endereço que será movido, e através da opção *Blocks/Move* é disponibilizada uma lista de blocos associados a esse nó, basta selecionar entre eles qual será movido pra outro nó. A segunda opção só é ativada no momento de remoção de um nó, caso este tenha blocos associados, a interface ilustrada pela Figura 41 é disponibilizada ao administrador.

A Figura 49 ilustra os procedimentos necessários para mover o bloco de endereço *200.136.30.0*, com máscara de rede *27*, alocado ao nó *São Paulo*, nó de origem do bloco, para o nó *Paraná*, selecionado para receber como nó de destino do bloco de endereço acima mencionado..

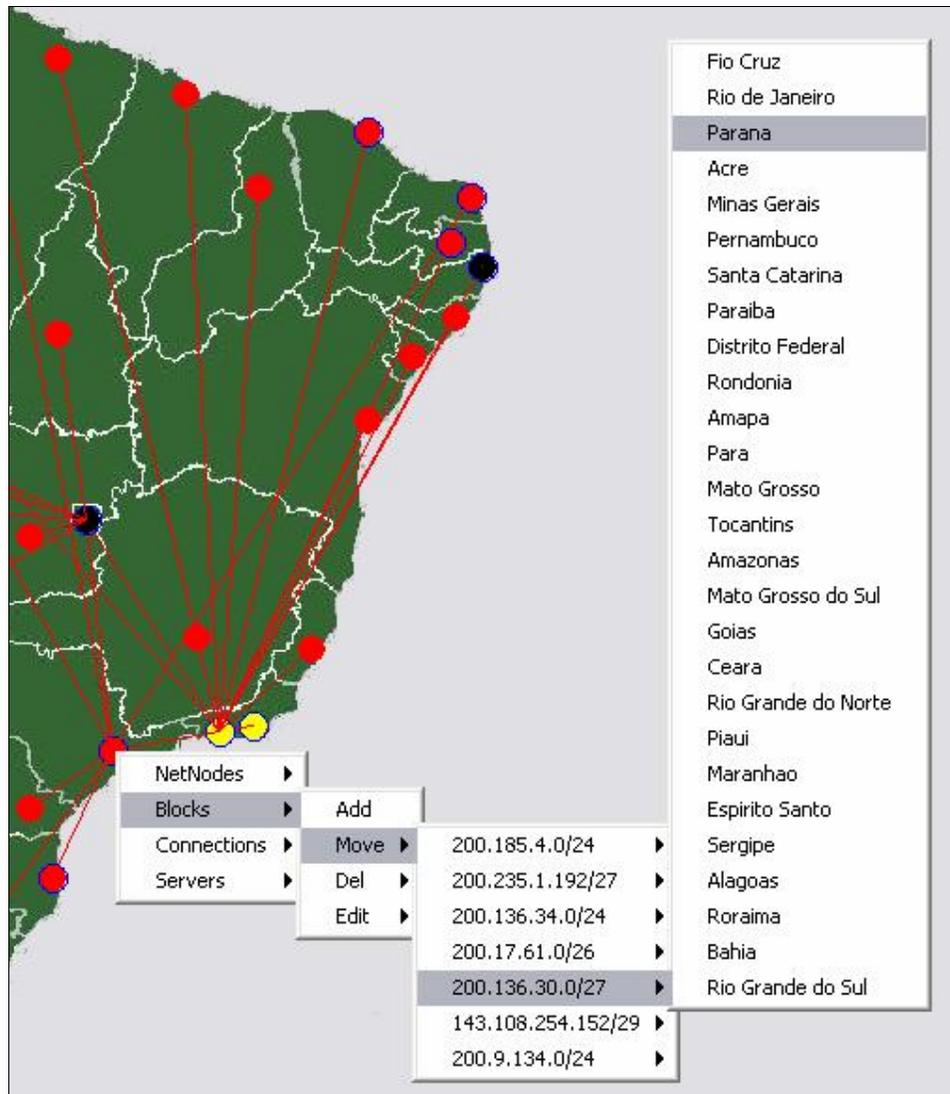


Figura 49 – Movendo um bloco de endereço IP para outro nó

5.2.7 – Configuração de uma conexão entre dois nós

O administrador dispõe de uma funcionalidade que o permite adicionar novas conexões entre dois nós existentes no documento de configuração. Para ativar essa funcionalidade, o administrador deve clicar com o botão direito do mouse sobre o nó que irá originar a conexão. Selecionando a opção *Connections/Add*, é disponibilizada a lista de nós que podem ter conexão com esse nó. Um nó origem não pode ter conexão com ele mesmo, bem como com outro nó no qual ele já possui conexão. A Figura 50 ilustra os procedimentos necessários para ativar essa funcionalidade.

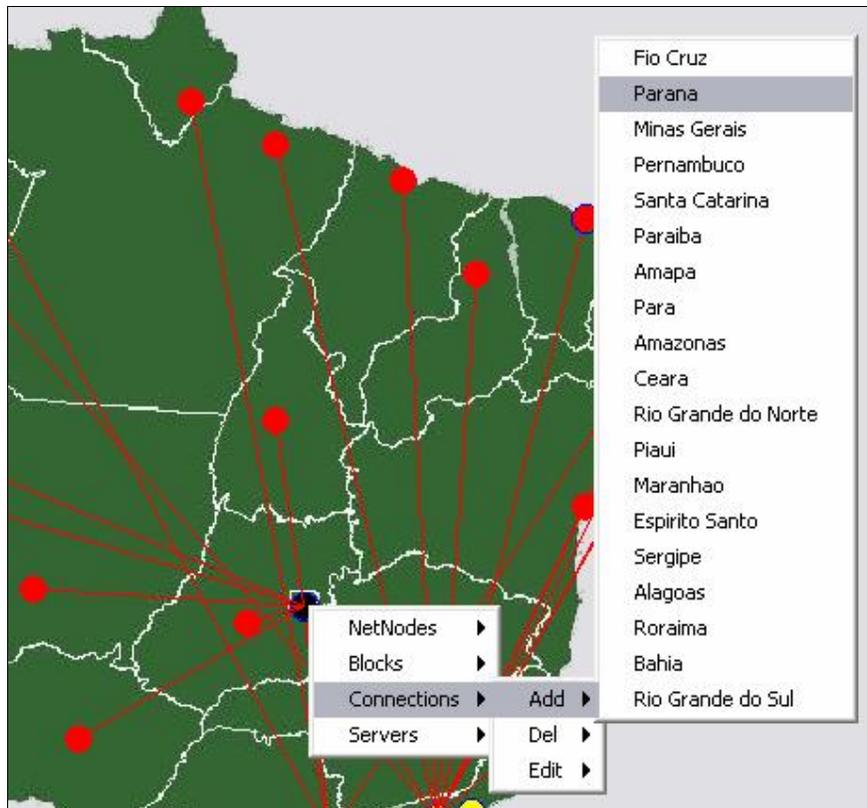


Figura 50 – Procedimento para inclusão de uma Conexão.

Como ilustrado pela Figura 50, o nó *Distrito Federal* foi selecionado como origem de uma conexão e o nó *Paraná*, selecionado como destino da mesma. Vale ressaltar que os nós que já possuem conexão com o nó *Distrito Federal* não foram listados.

Após selecionar os nós de origem e destino, a interface ilustrada pela Figura 51 é disponibilizada ao administrador para que este possa informar a taxa, bem como a unidade (*bps*, *Mbps*, *Gbps* e *Tbps*) dessa conexão. O campo *Source* descreve o nó de origem da conexão (*Distrito Federal*), enquanto o campo *Target* (*Paraná*) descreve o nó de destino da mesma. *Rate* (155) e *Unit* (*Mbps*) descrevem a velocidade dessa conexão e a unidade, respectivamente.



Figura 51 – Inclusão de uma conexão entre dois nós

O administrador dispõe de uma funcionalidade que o permite remover uma conexão entre dois nós existentes no documento de configuração. Para ativar essa funcionalidade, este deve clicar com o botão direito do mouse sobre o nó de origem ou destino da conexão e através da opção *Connections/Del*, selecionar em uma lista o nó que se deseja remover. Vale ressaltar que essa lista de nós, diferentemente do que acontece na adição de uma conexão (a lista é composta dos nós que não estão diretamente ligados a ele), é composta obviamente apenas pelos nós que possuem conexão com o nó de origem da mesma.

A Figura 52 ilustra os procedimentos necessários para ativar a interface de remoção de uma conexão. Neste exemplo, o administrador irá remover uma conexão existente entre o nó *São Paulo* (origem da conexão) e o nó *Paraná* (destino da conexão).

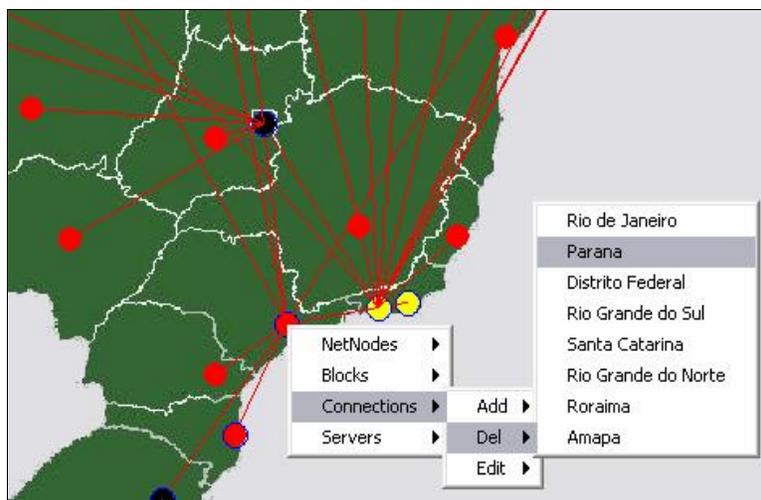


Figura 52 – Remoção de uma conexão entre dois nós

A edição da velocidade ou unidade de uma conexão pode ser realizada pelo administrador. Para isso deve seguir procedimentos análogos à adição de uma conexão entre dois nós existentes no documento de configuração. A opção *Connections/Edit* permite a alteração da taxa, bem como da unidade dessa conexão.

A Figura 53 ilustra os procedimentos necessários para ativar a interface de edição de uma conexão. O administrador pretende editar a conexão entre o nó *Rio de Janeiro* (origem da conexão), e o nó *Distrito Federal* (destino da conexão). Vale ressaltar que através da opção *Connections/Edit* é disponibilizada uma lista contendo apenas os nós que possuem conexão com o nó *Rio de Janeiro*. Após selecionar o nó de origem (*Rio de Janeiro*), bem como o nó de destino (*Distrito Federal*), a interface ilustrada pela Figura 54 é apresentada ao administrador contendo as informações da conexão entre esses dois nós.

O campo *Source* descreve o nome do nó de origem da conexão, enquanto o campo *Target* descreve o destino dessa conexão. Esta possui uma taxa descrita pelo campo *Rate* de 155 e a unidade *Mbps* descrita pelo campo *Unit*.

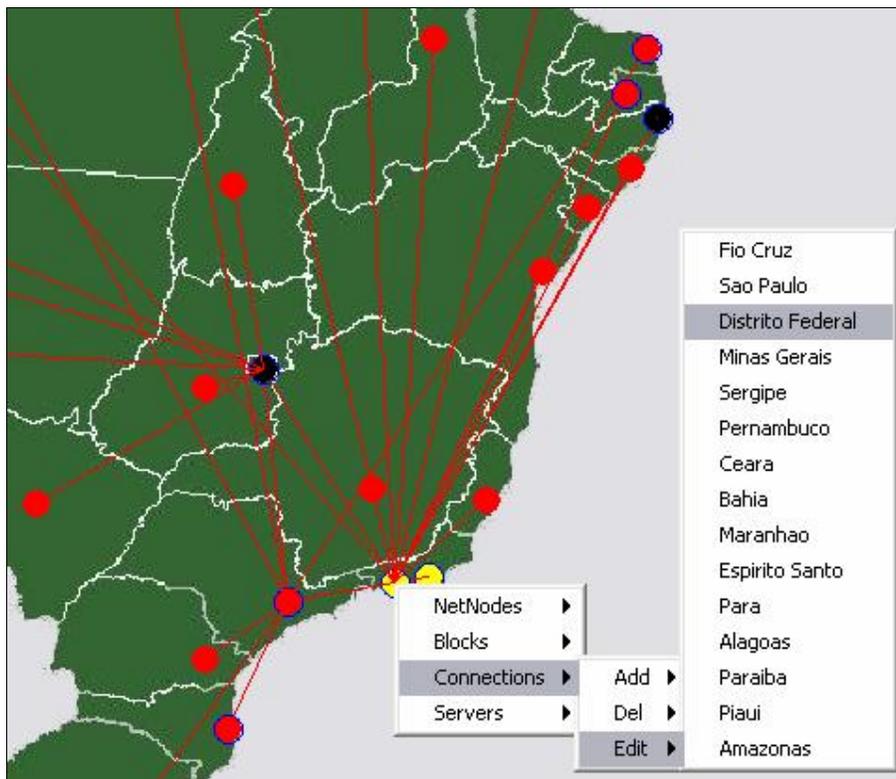


Figura 53 – Procedimentos para editar uma conexão entre dois nós



Figura 54 – Edição de uma conexão ente dois nós

5.2.8 – Configuração de Servidores

Servidores e seus respectivos serviços somente podem ser associados a nós previamente definidos no documento de configuração. Para inserir um novo servidor em um determinado nó, o administrador deve clicar o botão direito sobre o nó desejado e selecionar a opção *Servers/Add*. Dessa forma, a interface de adição de servidores é ativada. Esse procedimento pode ser visualizado através da Figura 55.

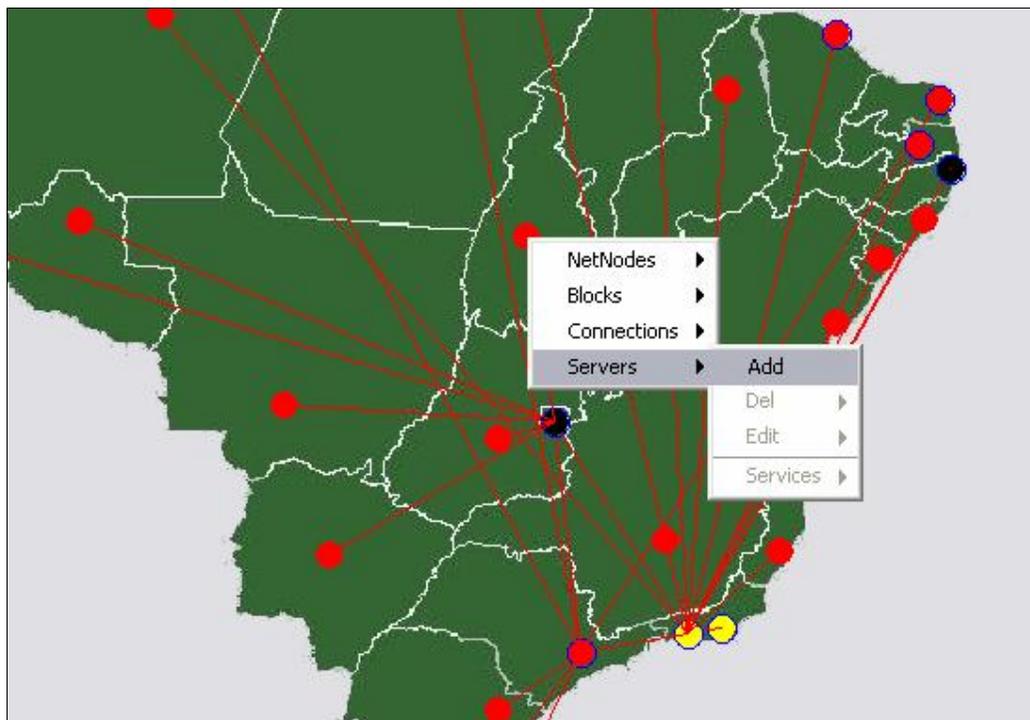


Figura 55 – Inserção de um servidor

Como pode ser observado, o procedimento de inserção de servidor também permite definir os serviços associados a ele. A opção que insere os serviços a um servidor se torna disponível apenas em nós que já possui servidores associados. Após selecionar a opção *Servers/Add*, a interface que permite a inserção de um novo servidor é disponibilizada ao administrador. Essa interface pode ser visualizada na Figura 56.

O campo *Node* corresponde ao nome do nó que terá um servidor associado a ele. *IP Address* descreve o endereço IP em que este servidor receberá requisições dos clientes em uso do sistema. Para que esse endereço IP seja válido, este deve estar inserido em um dos blocos alocados ao nó que irá receber o servidor. Para isso o configurador realiza a operação lógica *and* entre o endereço IP do servidor e a máscara de rede dos blocos contidos no documento de configuração alocados ao nó em questão. O resultado de cada operação lógica é comparado com os blocos de endereços descritos no documento de configuração. Se o resultado da operação é igual ao endereço base de algum bloco, então o endereço deste servidor pertence à respectiva rede e pode ser inserido no documento de configuração. As opções *Online* e *Default* representam o status desse servidor (*online/offline*) e se o mesmo é um servidor *default* no sistema de distribuição de vídeo.



Figura 56 – Inserção de um novo servidor

Como ilustrado pela Figura 56, será inserido um servidor ao nó *Paraná*, com endereço IP *150.132.165.1*. Este servidor está definido com *status Online* e não está configurado como servidor *default*.

Um serviço só pode ser associado a servidores, ou seja, a nós que possuem servidores associados a ele. Um nó pode ter vários servidores com diferentes serviços. O

administrador do sistema pode adicionar servidores e serviços *dvod* e *dlive*, representando os serviços de vídeo sob demanda e ao vivo, respectivamente, somente a nós e servidores anteriormente definidos. Todos os procedimentos para associar um serviço a um servidor serão descritos posteriormente.

Um servidor pode ser removido do documento de configuração, clicando com o botão direito do mouse sobre o nó que contém o servidor que se deseja remover. A opção *Servers/Del* disponibiliza a lista de servidores associados a esse nó.

Esse procedimento pode ser visualizado na Figura 57, onde o nó *Distrito Federal* possui dois servidores associados a ele, *200.19.119.112* e *200.150.119.60*. O servidor com endereço IP *200.150.119.60* foi selecionado para ser removido do documento de configuração. Vale ressaltar que todos os serviços associados a esse servidor também é completamente removido do documento.

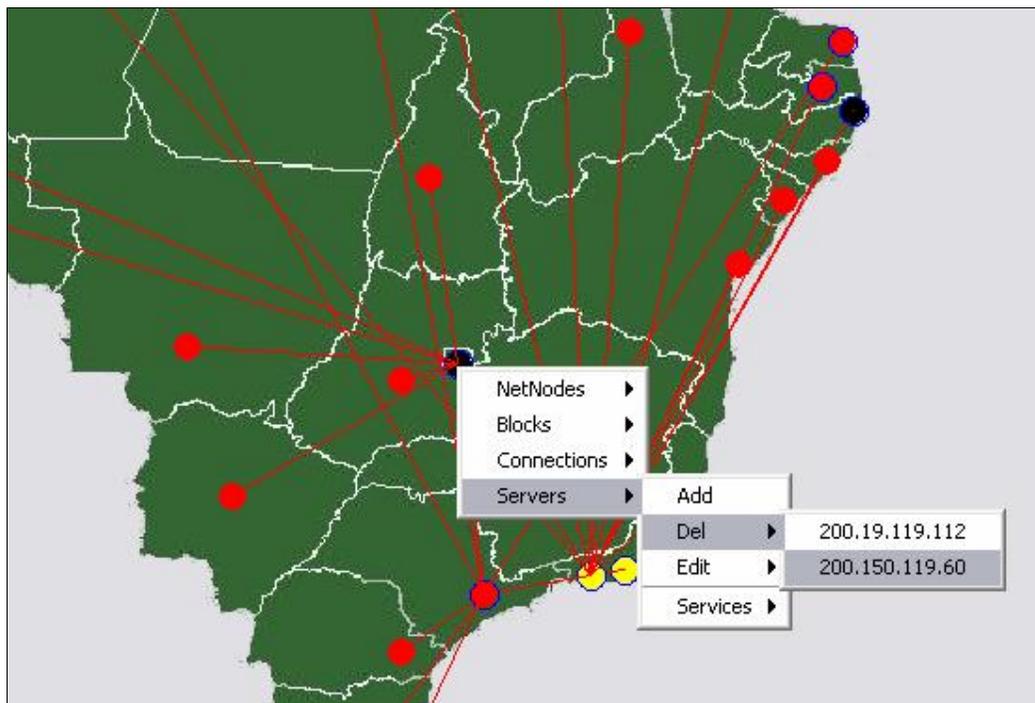


Figura 57 – Remoção de servidores

A funcionalidade de edição de servidores e serviços permite ao administrador pode editar informações referentes aos servidores e serviços existentes no documento de configuração. Nessa seção será descrito apenas os procedimentos necessários para edição de informações referentes a servidores, os serviços serão descritos posteriormente. Seguindo os procedimentos análogos a remoção de um servidor, ilustrado pela Figura 57, o administrador pode através da opção *Servers/Edit* selecionar

através da lista de servidores disponíveis nesse nó, o servidor que se deseja realizar a edição das informações. A Figura 58 ilustra a interface disponibilizada para a edição de um servidor, bem como suas respectivas informações.

Vale ressaltar que, para que o endereço IP descrito no campo *IP Address* seja válido, este deve estar inserido em um dos blocos alocados ao nó descrito pelo campo *Node* que receberá o servidor. Para isso o configurador realiza os mesmos procedimentos descritos na inserção de um servidor.



Figura 58 – Edição de um servidor

No exemplo ilustrado pela Figura 58, o campo *Node* representa o nome do nó que contém o servidor (*Paraiba*), o campo *IP Address*, descreve o endereço IP que este servidor receberá requisições (*150.165.132.1*). O campo *status* descreve o estado atual desse servidor (*Online*) e a opção *default* descreve que este servidor não está definido como *default* (padrão no sistema).

5.2.8.1 – Consulta manual do servidor *default*

O configurador provê mecanismos que permitem ao administrador do serviço de distribuição de vídeo obter informações sobre os nós e identificar o estado operacional dos servidores, bem como o servidor *default* existente na rede de distribuição. O estado ativo/inativo dos servidores, e a definição de quem é o servidor *default* já são explicitamente representados na interface principal adotando diferentes cores. Entretanto, como complemento, o configurador também oferece uma alternativa para a obtenção manual de quem está definido como servidor *default* disponível no sistema de

distribuição, bem como sua modificação. Os procedimentos para consultar manualmente quem está definido como servidor *default* pode ser visualizado através da Figura 59.

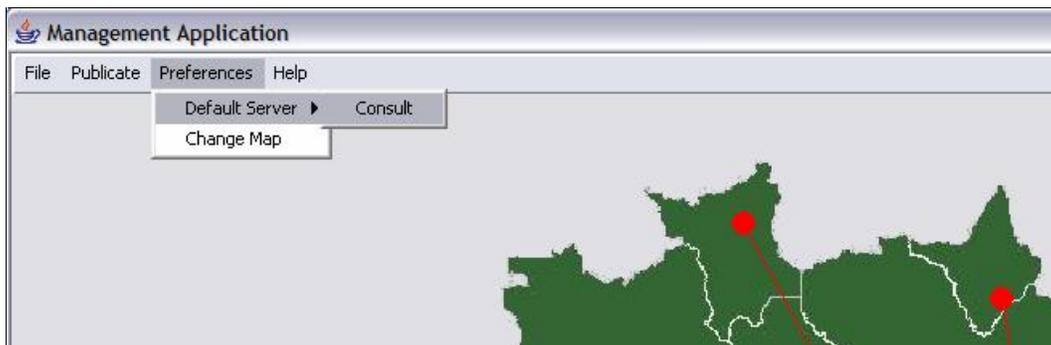


Figura 59 – Consulta manual do servidor *default*

Após selecionar a opção *Preferences/Default Server/ Consult*, é disponibilizada ao administrador uma interface que contém o nome do servidor *default* disponível no sistema de distribuição de vídeo. Esta interface pode ser visualizada na Figura 60.



Figura 60 – Informação manual do servidor *default*

5.2.8.2 – Configuração de um serviço

Para associar um serviço a um determinado servidor, o administrador deve clicar o botão direito sobre o nó desejado, selecionar um servidor disponível nesse nó e, através da opção *Services/<Lista de servidores>/Add*, adicionar um novo serviço no servidor anteriormente selecionado. A Figura 61 ilustra os procedimentos necessários para inserção de um novo serviço no documento de configuração.

O nó *Distrito Federal* dispõe de um servidor associado a ele com endereço IP *200.19.119.112*. Dessa forma, deseja-se então adicionar um novo serviço a esse servidor. A interface que permite adição de um novo serviço em um servidor previamente selecionado pode ser visualizada através da Figura 62.

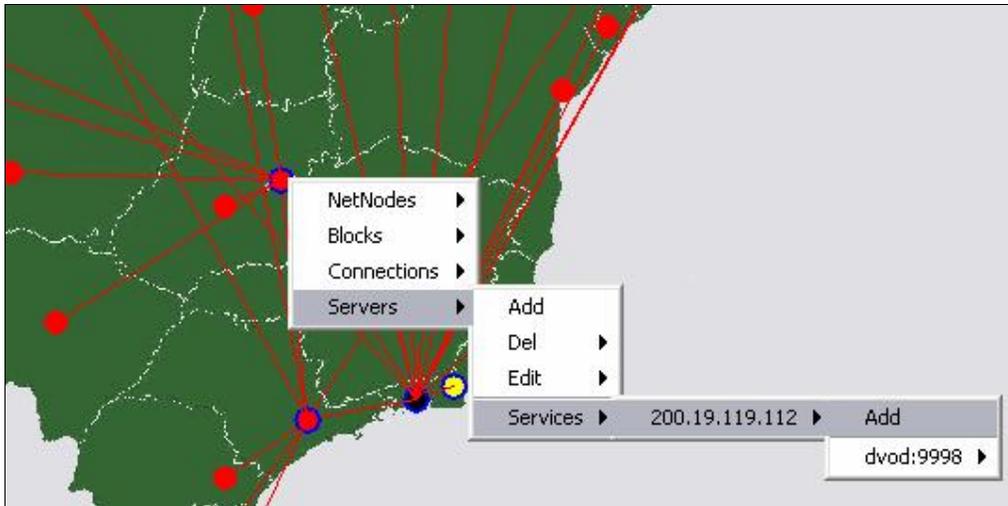


Figura 61 – Inserção de um serviço

No exemplo da Figura 62, o administrador está inserido um serviço ao vivo (*dlive*), rodando na porta 6500, no servidor *200.19.119.112*, associado ao nó *Distrito Federal*.



Figura 62 – Interface de inserção de um novo serviço

O configurador dispõe de mecanismos que permite ao administrador editar serviços associados aos servidores disponíveis no documento de configuração. Através da opção *Servers/Services/<Lista de servidores>/<Lista de serviços>/Edit*, pode-se então editar o serviço selecionado. A Figura 63 ilustra os procedimentos necessários para editar um serviço associado a um servidor, disponível no documento de configuração.

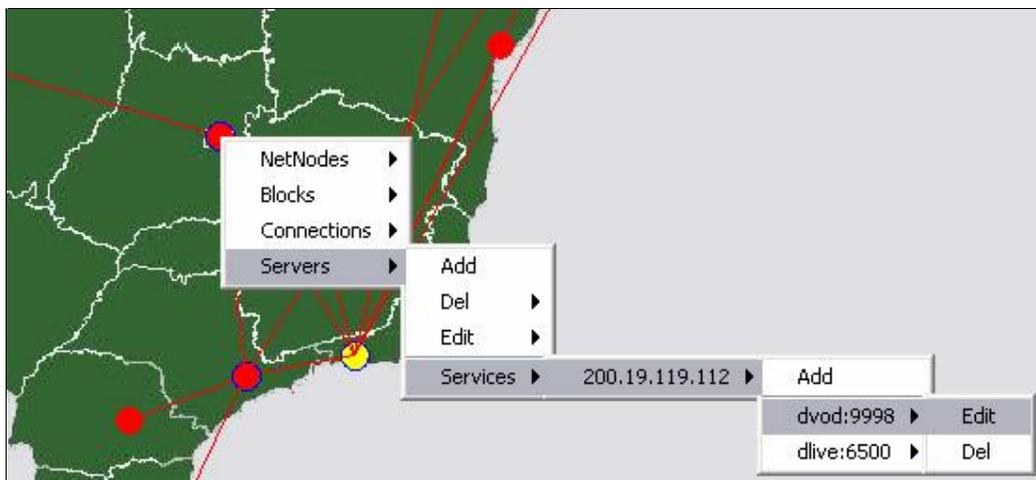


Figura 63 – Edição de um serviço

O nó *Distrito Federal* dispõe de um servidor com endereço IP *200.19.119.112*, o qual possui dois serviços associados a ele, um serviço sob demanda (*dvod*), rodando na porta 9998, e um serviço ao vivo (*dlive*), rodando na porta 6500. Após selecionar o serviço que será editado, a interface ilustrada anteriormente pela Figura 62 será disponibilizada ao administrador e este estará apto a realizar a alteração no serviço selecionado.

Para remover um serviço associado a um servidor, o administrador deve seguir os procedimentos análogos aos utilizados na edição de um serviço, ilustrado pela Figura 63. A opção *Servers/Services/<Lista de servidores>/<Lista de serviços>/Del*, permite a remoção de um serviço associado ao servidor previamente selecionado.

5.2.9 – Salvar informações

Para salvar as alterações realizadas nas informações contidas no documento de configuração, o administrador deve fazer uso da opção *File/Save*, contida na barra de menus do configurador. Esse procedimento pode ser visualizado na Figura 64.

Como já mencionado anteriormente, as modificações são realizadas localmente. Dessa forma, para tornar essas informações públicas ao coordenador, estas devem ser enviadas ao mesmo. Os procedimentos de *download* e *upload* do documento de configuração podem ser visualizados no item 5.2.2 na Figura 34.



Figura 64 – Salvando as novas informações no documento de configuração

5.2.10 – Trocar o cenário do Configurador

O administrador do sistema dispõe de uma funcionalidade importante que permite a troca do cenário utilizado pelo coordenador. Atualmente o configurador dispõe de um cenário contendo o mapa do Brasil. Vale ressaltar que a troca deste cenário, implica na obtenção de uma imagem geoprocessada, bem como o conhecimento das coordenadas geográficas da mesma. A Figura 65, ilustra os procedimentos necessários para ativar a funcionalidade de troca de cenário da aplicação.

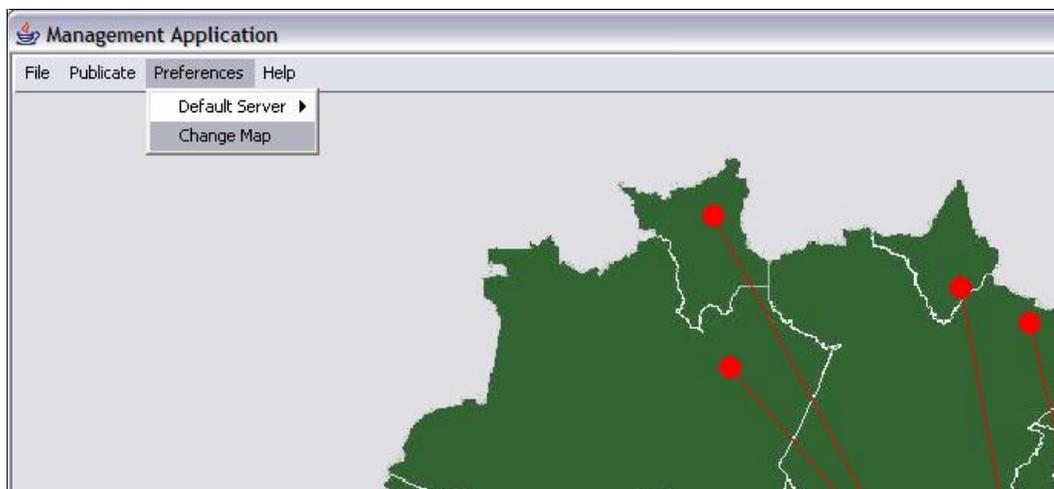
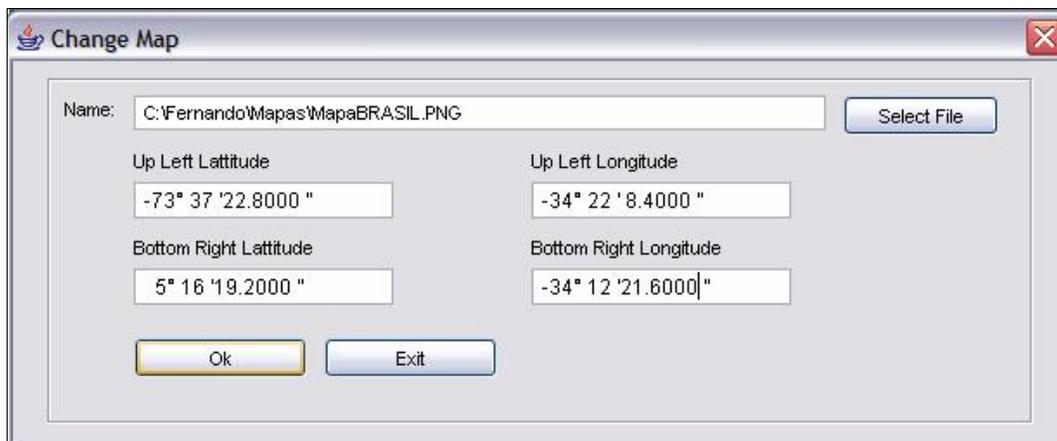


Figura 65 – Procedimentos para troca de cenário

De acordo com a Figura 65, através da opção *Preferences/Change Map* o administrador pode ativar a interface responsável pela troca de cenário. A Figura 66 ilustra a interface que permite a troca de cenário utilizado pelo configurador.



5.2.12 – Considerações Finais

Este capítulo apresentou a Aplicação de Configuração (Configurador) adotada para a manipulação das informações contidas no documento de configuração do serviço de distribuição de vídeo digital.

Capítulo 6

Estudo de Caso

Esta seção descreve estudos de casos onde foram realizados experimentos reais utilizando a infraestrutura proposta. No Cenário 1 será descrito um experimento nacional com transmissão ao vivo. Posteriormente, no Cenário 2, descrevemos um experimento internacional no qual envolve transmissão ao vivo entre Brasil e Estados Unidos.

6.1 – Cenário 1: Transmissão SBRC e WRNP 2004

Esse estudo de caso descreve um experimento nacional para transmissão ao vivo do Workshop da RNP, realizado durante o SBRC 2004 em Gramado (RS). Neste experimento, o sinal de vídeo foi gerado por uma câmera SDTV (*Standard Definition TV*) conectada a um codificador. O servidor *SBRC* atua como a fonte do fluxo de vídeo para os demais servidores intermediários disponíveis na rede de distribuição. A transmissão do fluxo de vídeo do codificador até o servidor *SBRC* é feita pelo protocolo TCP. Vale salientar que o codificador e o servidor *D-Live* estão conectados a uma mesma rede. A Figura 68 ilustra a arquitetura de interconexão utilizada para essa transmissão.

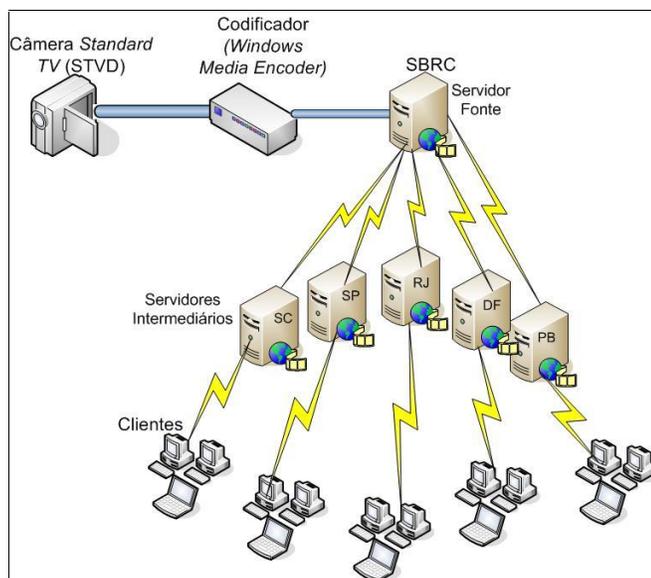


Figura 68 – Arquitetura do Experimento SBRC/WRNP 2004

Devido à limitação de espaço e para melhor compreensão da proposta apresentada, iremos descrever este cenário utilizando apenas um subconjunto das informações contidas no documento de configuração referentes à rede de distribuição da RNP, dentre eles estão os nós: *Santa Catarina, Paraná, São Paulo, Rio de Janeiro, Minas Gerais, Acre, Distrito Federal, Bahia, Pernambuco, Paraíba, Rio Grande do Norte e Ceará.*

Partindo do princípio que um servidor deve ser instalado em um nó existente no documento de configuração, o administrador deve então verificar a existência desse nó antes de adicionar um servidor a ele. Suponhamos que o nó SBRC (servidor fonte para os demais servidores) não exista no documento de configuração. Para que este nó seja adicionado no documento, o administrador deve, através do Configurador ilustrado pela Figura 69, ativar a interface responsável pela inserção de um nó no documento de configuração.

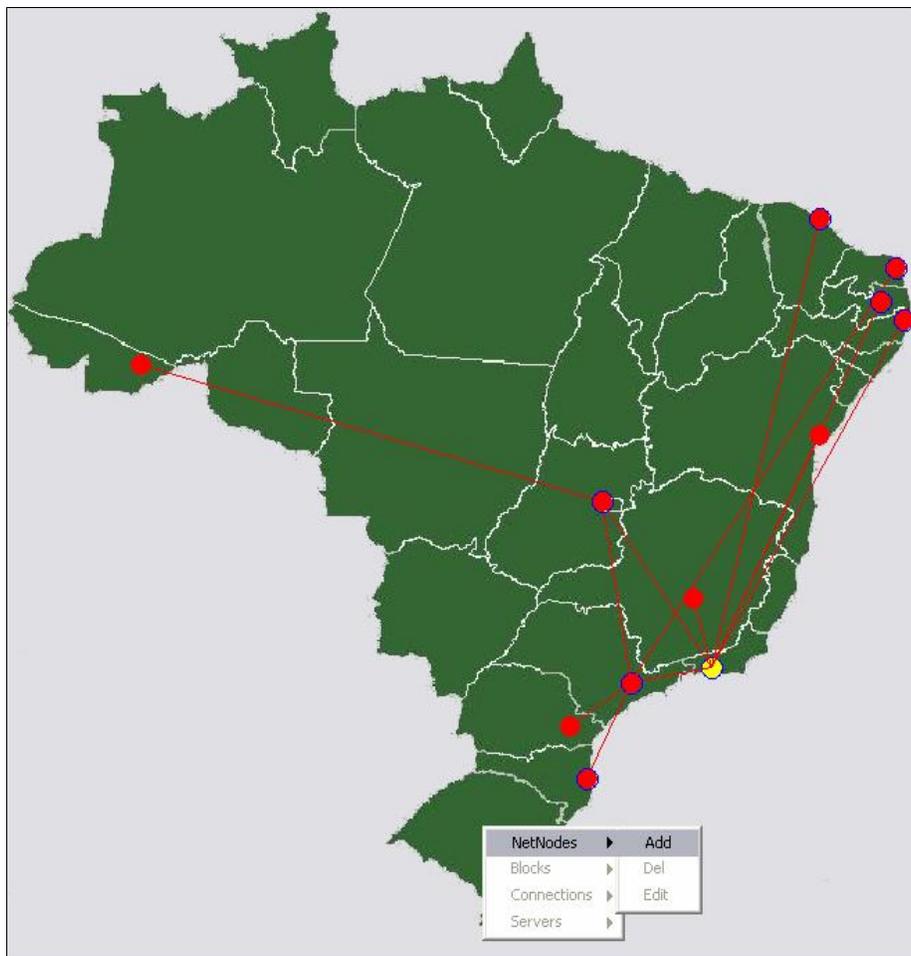
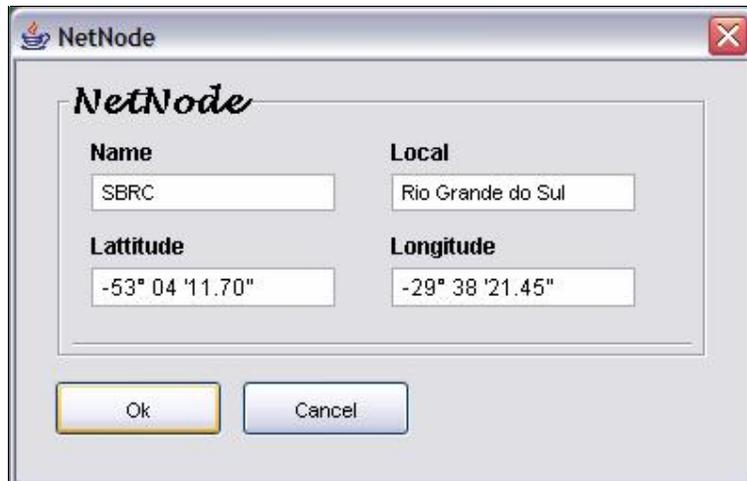


Figura 69 – Adição de um nó

A opção *Netnodes/Add* ativa a interface de inserção de um nó no documento de configuração. Para o cenário em questão, será adicionado um nó denominado *SBRC*, localizado no *Rio Grande do Sul*, através das coordenadas geográficas $-54^{\circ} 04' 11.70''$ e $-29^{\circ} 38' 21.45''$. Essa interface pode ser visualizada através da Figura 70.



The image shows a dialog box titled "NetNode" with a close button in the top right corner. Inside the dialog, there are four input fields arranged in a 2x2 grid. The top-left field is labeled "Name" and contains the text "SBRC". The top-right field is labeled "Local" and contains the text "Rio Grande do Sul". The bottom-left field is labeled "Latitude" and contains the text "-53° 04 '11.70\"". The bottom-right field is labeled "Longitude" and contains the text "-29° 38 '21.45\"". At the bottom of the dialog, there are two buttons: "Ok" on the left and "Cancel" on the right.

Figura 70 – Inserção do nó SBRC

Após a inserção do nó *SBRC*, um servidor pode ser associado a ele. A Figura 71 ilustra os procedimentos necessários para inserção de um servidor no documento de configuração.

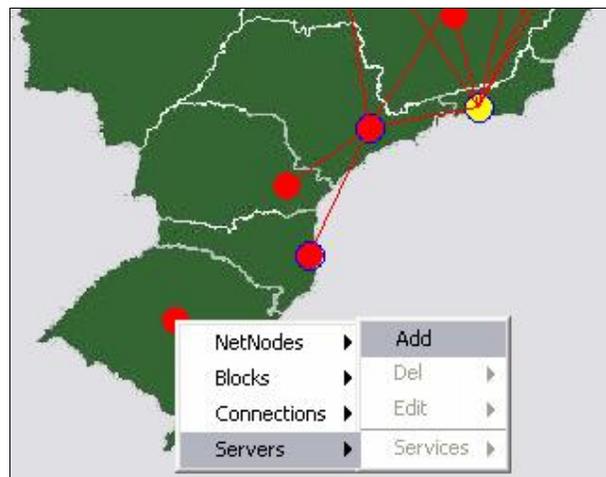


Figura 71 – Inserção de um servidor ao nó SBRC

Após selecionar a opção *Servers/Add*, a interface responsável pela inserção de um servidor é disponibilizada ao administrador e pode ser visualizada na Figura 72.

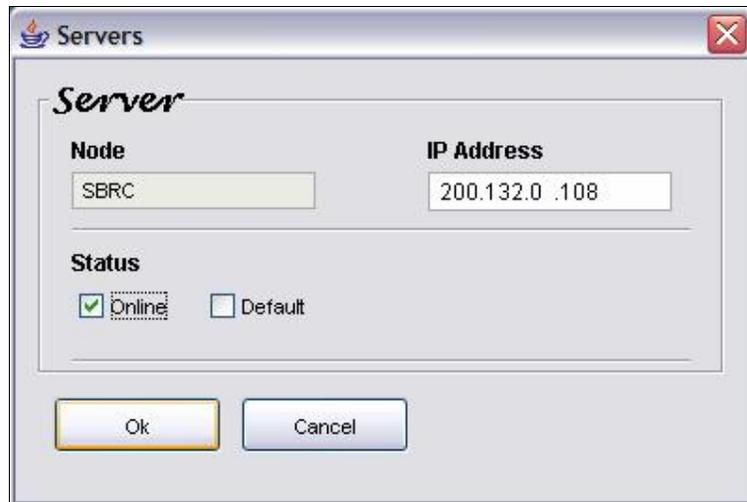


Figura 72 – Inserção do servidor RS no documento de configuração

O nó *SBRC* possui um servidor associado que receberá requisições através do endereço IP *200.132.0.108*. Posteriormente à inserção de um novo servidor no documento de configuração, o administrador deve associar os tipos de serviços que este servidor disponibilizará aos clientes (*dvod* ou *dlive*), bem como a porta em que este servidor receberá requisições.

Os procedimentos necessários utilizados para associar um ou vários serviços ao servidor *SBRC*, pode ser visualizado na Figura 73. Será associado um serviço ao servidor *200.132.0.108*, alocado ao nó *SBRC*.

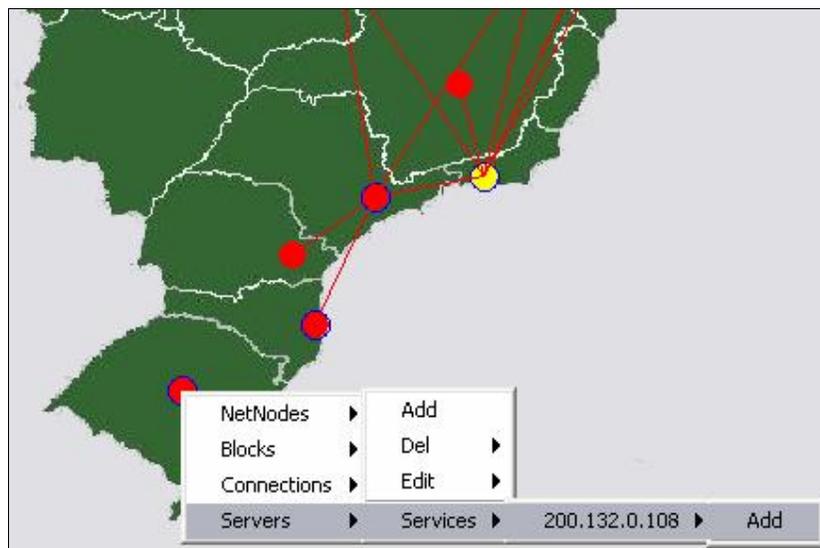


Figura 73 – Procedimentos para inserção de um serviço ao nó *SBRC*

A Figura 74(a) ilustra a interface ativada para a inserção de um novo serviço sob demanda (*dvod*), enquanto que a Figura 74(b) ilustra a inserção de um serviço ao vivo (*dlive*) ao servidor *200.132.0.108*.



Figura 74 – Inserção de serviços ao servidor *200.132.0.108* – *SBRC*

Para finalizar a configuração do nó *SBRC* é preciso conectá-lo a outros nós. Respeitando as diretrizes, normas e estrutura da rede de distribuição RNP, o nó *SBRC* (representado pelo POP Rio Grande do Sul) possui ligação com o nó *São Paulo* (representado pelo POP São Paulo). Para ativar a interface que permite o administrador inserir uma conexão entre dois nós, os procedimentos ilustrados pela Figura 75 deverão ser efetuados. O nó *Distrito Federal* dispõe de um servidor associado a ele com endereço IP *200.19.119.112*. Dessa forma, deseja-se então adicionar um novo serviço a esse servidor. A interface que permite a adição de um novo serviço em um servidor previamente selecionado pode ser visualizada através da Figura 62.



Figura 75 – Inserindo uma conexão entre o nó *São Paulo* e *SBRC*

Após realizar os procedimentos ilustrados na Figura 75, a interface responsável pela inserção de uma conexão entre dois nós pode ser visualizada na Figura 76.

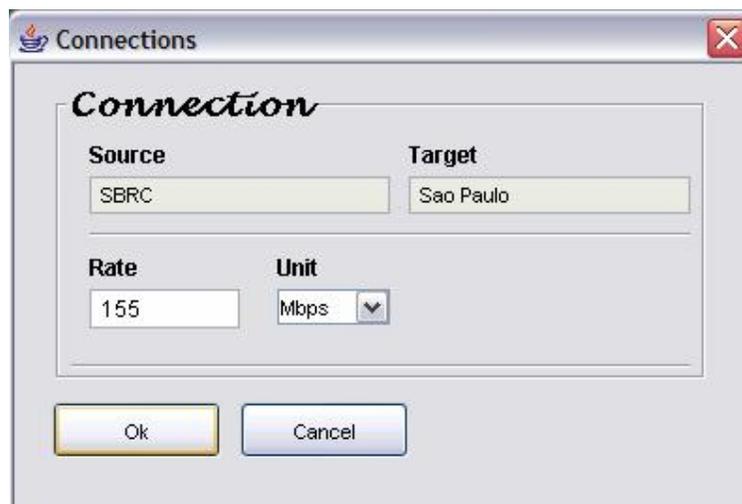


Figura 76 – Inserção de uma conexão entre os nós *SP* e *SBRC*

Como ilustrado pela Figura 76, está sendo inserida uma conexão entre os nós *São Paulo* e *SBRC*, com uma taxa de conexão de *155 Mbps*.

Para inserir blocos de endereços IP a um nó *SBRC*, o administrador deve seguir os procedimentos ilustrados na Figura 77.

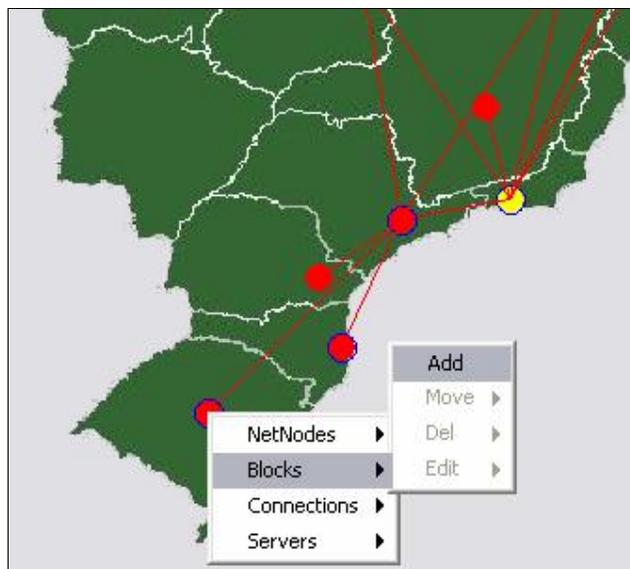


Figura 77 – Procedimentos utilizados na inserção de blocos ao nó *SBRC*

Após seguir os procedimentos ilustrados pela Figura 77, a interface que permite a inserção de blocos aos nós é disponibilizada ao administrador do sistema. Essa interface pode ser visualizada através da Figura 78.

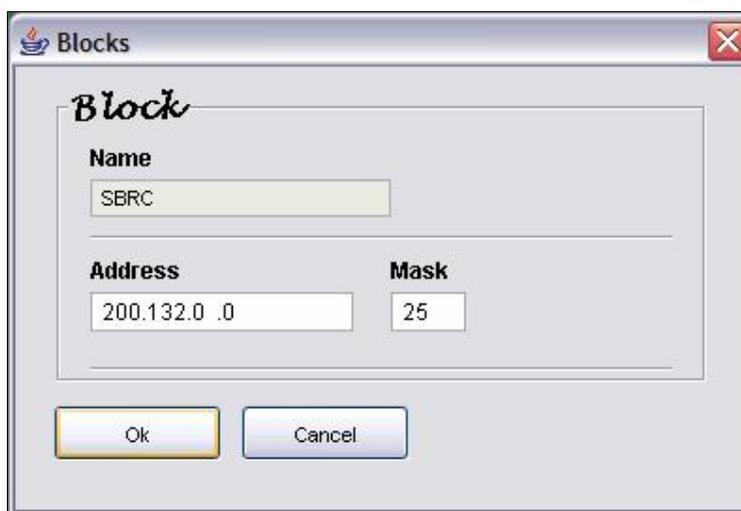


Figura 78 – Inserção de blocos ao nó *SBRC*

Dessa forma, será inserido o bloco de endereço IP *200.132.0.0*, com máscara de rede *25* ao nó *SBRC*.

Ao efetuar a configuração do serviço como ilustrado pelas Figuras 69 a 78, para que essas informações sejam adicionadas no documento de configuração, elas devem ser salvas utilizando os mesmo procedimentos ilustrados na Figura 64. Para melhor compreensão de como essas informações são descritas nesse documento, a Figura 79 ilustra os trechos XML do documento de configuração, que foi manipulado pelo

configurador durante o experimento. Para realizar a configuração do experimento, inicialmente, foi necessário incluir no documento de configuração os nós *SBRC* e *Sao Paulo*, e, em seguida, criar uma conexão entre os mesmos. Para representar os clientes locais, o bloco *200.132.0.0/25* foi associado ao nó *SBRC*. Por fim, o servidor *200.132.0.108*, um serviço *dlive* e um *dvod* foram adicionados ao mesmo.

```
<netdescr">
<netnodes>
...
<node name="SBRC" local="Rio Grande do Sul" latitude="-29.6993789954"
      longitude="-52.7711796043" default="no" hasServer="yes" />
</netnodes>

<connections>
...
<source name="SBRC">
<target name="Sao Paulo" rate="155" unit="Mbps" />
</node>
</connections>

<blocks>
...
<node name="SBRC">
<block address="200.132.0.0" mask="25" />
</node>
</blocks>
<poll>
...
<node name="SBRC" status="online">
<server ip="200.132.0.108">
<service name="dvod" port="9998" />
<service name="dlive" port="6500" />
</server>
</node>
</poll>
</netdescr">
```

Figura 79 – Configuração SBRC/WRNP 2004

Para que possa haver uma conexão entre dois nós, estes devem ser definidos como elementos *<netnodes>*.

Neste cenário o servidor *SBRC* recebe requisições para transmissões ao vivo (*d-live*) através do endereço IP *200.132.0.108*, porta *6500* e transmissões sob demanda através da porta *9998*. Foi definido como produtor de fluxo (servidor fonte), enquanto o servidor *São Paulo*, serve de ponte (intermediário) para que o servidor *Rio de Janeiro* distribua o fluxo aos demais servidores instalados no serviço de distribuição. Os clientes pertencentes ao bloco *200.132.0.0*, máscara *25*, podem fazer uso do sistema e receber os

fluxos de vídeo gerado diretamente pelo servidor fonte, enquanto os demais clientes recebem fluxos dos servidores intermediários mais próximos deles.

Para ilustrar um subconjunto das informações existentes no documento de configuração, outros servidores foram instalados, bem como as conexões entre eles e as sub-redes que estes irão atender.

Neste cenário, o servidor associado ao nó *SBRC* foi definido como produtor do fluxo de vídeo (servidor fonte). Este servidor recebe requisições para transmissões ao vivo (*d-live*). Por outro lado, o servidor associado ao nó *São Paulo* atua como intermediário para os demais servidores instalados na rede de distribuição (*Santa Catarina, Rio de Janeiro, Distrito Federal, Bahia, Pernambuco, Paraíba, Rio Grande do Norte e Ceará*). Os clientes locais recebem o fluxo de vídeo gerado diretamente pelo servidor fonte. Por outro lado, os demais clientes recebem fluxos através dos servidores intermediários mais próximos deles.

Para melhor compreensão, serão apresentados apenas três clientes fictícios, que solicitam vídeo aos servidores mais próximo deles. A Figura 80 ilustra a geração de fluxo pelo servidor *SBRC*, bem como o uso dos servidores intermediários *SP, RJ e DF*, destinados a atender as requisições dos clientes 1, 2 e 3, respectivamente. No entanto, a Figura 80 tem objetivo apenas ilustrativo do cenário de uso em questão, não sendo gerada pelo Configurator.

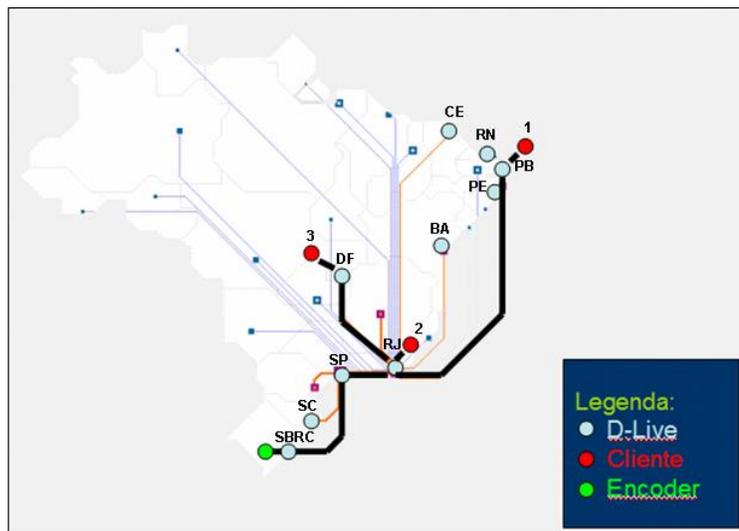


Figura 80 – Transmissão ao vivo do PoP – RS

Juntamente ao servidor *SBRC* um *Encoder* foi instalado para realizar as funções de codificação e compressão dos dados a serem transmitidos (ao vivo) na rede de distribuição.

O cliente 1 solicita o vídeo transmitido ao servidor PB, como este não possui uma cópia do vídeo requisitado, este servidor deverá solicitá-lo a quem possuir uma cópia do mesmo, sempre levando em consideração a velocidade do *link* (proximidade) entre eles. Neste cenário, uma requisição é realizada para o servidor RJ, SP e SBRC, respectivamente. Dessa forma o cliente 1 receberá o fluxo do vídeo transmitido pelo servidor SBRC (fonte) através dos servidores intermediários SP, RJ e PB.

Posteriormente, o cliente 2 requisita o vídeo transmitido ao servidor RJ, como este servidor possui uma cópia desse vídeo, o mesmo poderá atender a requisição feita por esse cliente através dos dados contidos em sua memória *cache*, aliviando assim a carga do servidor fonte. O cliente 3 pertence ao bloco de endereço alocado ao nó DF que, conseqüentemente, possui um servidor instalado nele. No momento em que este cliente solicita o vídeo ao servidor DF, como este não possui uma cópia do vídeo em questão, deverá solicitar ao servidor mais próximo a ele (que possua uma cópia do vídeo) e atender a requisição do cliente. Neste exemplo, o servidor RJ será o gerador de fluxo para o servidor DF, que posteriormente o retransmite ao cliente.

6.2 – Cenário 2: The 7th Annual SURA/ViDe Conference

Esse estudo de caso descreve um experimento internacional para transmissão ao vivo de vídeo de alta definição (MPEG-2) entre Brasil e Estados Unidos da conferência *SURA/ViDe*, realizada em Atlanta (EUA) em março de 2005. Neste experimento, o sinal de vídeo foi gerado por uma câmera HDTV (*High Definition TV*) conectada a um codificador MPEG-2. A saída do codificador alimenta o *TS Processor Tandberg*, que, por sua vez, encapsula o fluxo de vídeo em pacotes IP e os envia para o servidor *D-Live* através do protocolo UDP. O servidor *SuraVide* atua como a fonte do fluxo de vídeo para os demais servidores intermediários disponíveis na rede de distribuição.

Dois servidores foram utilizados na transmissão:

- Um servidor de vídeo para transmissões sob-demanda (*D-VoD*);
- Um servidor de vídeo para transmissões ao vivo (*D-Live*).

O servidor *D-Live* aguarda por requisições dos clientes que farão uso do serviço. Este servidor pode se conectar a outros servidores para otimizar o uso da rede, sendo cada um responsável em atender sub-redes distintas.

O servidor *D-VoD* recebe o fluxo de vídeo gerado como um cliente *D-Live*, e então o armazena, para posteriormente fornecer vídeo sob demanda aos clientes.

Baseado em parâmetros pré-definidos, este servidor armazena fluxos de vídeo em um arquivo, para posterior publicação e torna-os disponíveis através do protocolo HTTP.

Foram realizados dois testes: uma gravação de 4h30 de vídeo MPEG-2, com uma taxa de 6Mbps e outra gravação de 1h de vídeo MPEG-2, a uma taxa de 20Mbps.

Um servidor *D-VoD* pode atender a requisições dos clientes com fluxos de vídeo que se encontram em seus discos locais. A Figura 81 ilustra a arquitetura de interconexão utilizada para essa transmissão.

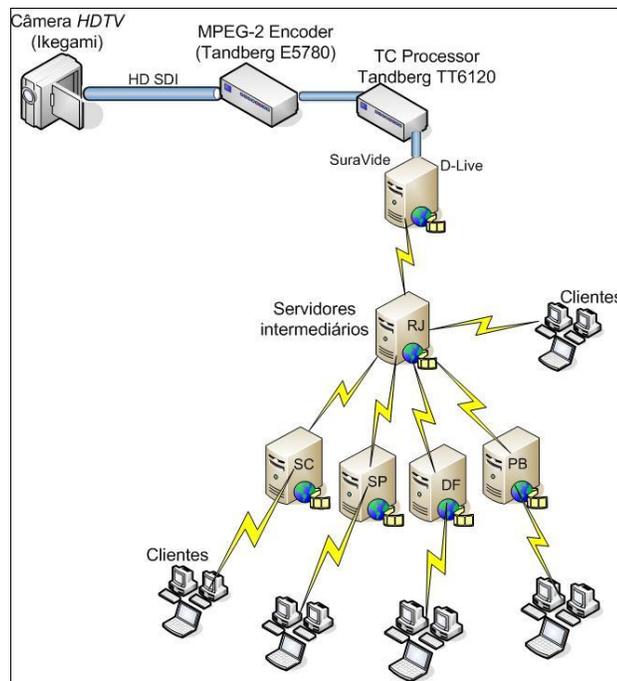


Figura 81 – Arquitetura do experimento realizado.

Utilizando os mesmos procedimentos ilustrados pela Figura 69, o administrador ativará a interface de inserção de um novo nó no documento de configuração. Para o cenário em questão, será adicionado um nó denominado *SuraVide*, localizado em *Atlanta – EUA*, através das coordenadas geográficas $33^{\circ} 45' 36''$ e $84^{\circ} 24' 00''$. A interface de inserção pode ser visualizada na Figura 82.

Após a inserção do nó *SuraVide*, um servidor pode ser associado a ele. A Figura 71 ilustra os procedimentos necessários para inserção de um servidor no documento de configuração. A interface responsável pela inserção de um servidor, pode ser visualizada na Figura 83.



Figura 82 – Inserção do nó *SuraVide*

A Figura 83 ilustra o nó *SuraVide* que possui um servidor associado o qual receberá requisições através do endereço IP *200.231.123.1*. Posteriormente a inserção de um novo servidor no documento de configuração, o administrador deve associar os tipos de serviços que este servidor disponibilizará aos clientes (*dvod* ou *dlive*), bem como a porta em que este servidor receberá requisições.



Figura 83 – Inserção do servidor *200.231.123.1* ao nó *SuraVide*

Como o servidor RJ possui o *link* de entrada e saída de dados no Brasil, o servidor instalado em *Atlanta* deve possuir uma ligação com este nó, e este se responsabiliza em distribuir o fluxo de vídeo aos demais servidores espalhados pelo País. A interface que permite a inserção de uma conexão entre o nó *SuraVide* e *Rio de Janeiro* pode ser visualizada na Figura 84.



Figura 84 – Inserção de uma conexão entre Brasil e EUA

Para inserir blocos de endereços IP ao nó *SuraVide*, o administrador deve seguir os procedimentos ilustrados na Figura 78. Após selecionar a opção que permite a inserção de blocos, a interface de inserção é disponibilizada ao administrador. A Figura 85 ilustra a interface que permite a inserção de blocos de endereços IP.

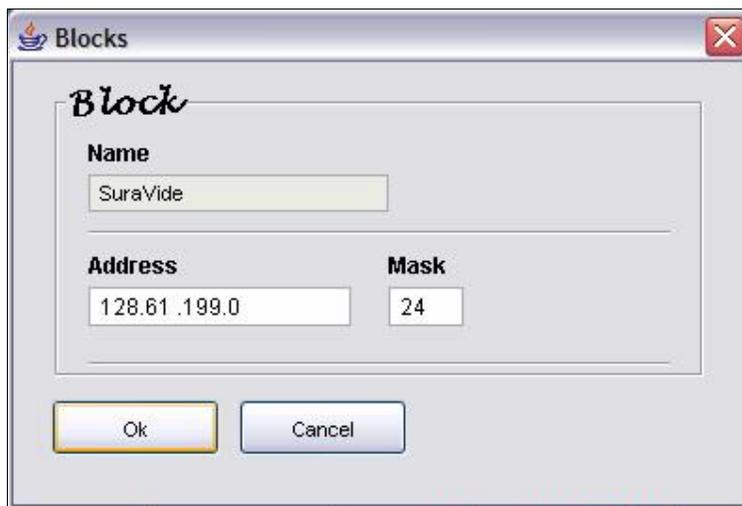


Figura 85 – Inserção do bloco 128.61.199.0 ao nó SuraVide

Para que essas informações sejam adicionadas no documento de configuração, elas devem ser salvas utilizando os mesmos procedimentos ilustrados na Figura 64. Para melhor compreensão de como essas informações são descritas nesse documento, a Figura 86 ilustra os trechos XML do documento de configuração, que foi manipulado pelo configurador durante o experimento. Para realizar a configuração do experimento, inicialmente, foi necessário incluir no documento de configuração o nó *SuraVide*, em seguida, criar uma conexão entre este nó e a porta de entrada no país que é o nó *Rio de Janeiro*. Para representar os clientes locais, o bloco *128.61.199.0/24* foi associado ao nó

SuraVide. Por fim, o servidor *SuraVide* possui um servidor, que recebe requisições através do endereço IP *128.61.199.14*. Os serviços *d-live* e *dvod* rodam nas portas *6500* e *9998*, respectivamente. Os clientes pertencentes à rede *128.61.199.0*, máscara *24*, podem fazer uso do sistema e receber os fluxos de vídeo gerado diretamente por esse servidor.

```
<netdescr">
<netnodes>
...
<node name="SuraVide" local="Atlanta" latitude="33,76"
      longitude="84,4" default="no" hasServer="yes" />
</netnodes>

<connections>
...
<source name="SuraVide">
<target name="Rio de Janeiro" rate="155" unit="Mbps" />
</node>
</connections>

<blocks>
...
<node name="SuraVide">
<block address="128.61.199.0" mask="24" />
</node>
</blocks>

<pool>
...
<node name="SuraVide" status="online">
<server ip="128.61.199.14">
<service name="dvod" port="9998" />
<service name="d-live" port="6500" />
</server>
</node>
</poll>
</netdescr>
```

Figura 86 – Configuração SuraVide 2005

Neste cenário o servidor associado ao nó *SuraVide* foi definido como produtor do fluxo de vídeo (servidor fonte). Este servidor recebe requisições para transmissões ao vivo (*d-live*). Por outro lado, o servidor associado ao nó *Rio de Janeiro* atua como intermediário para os demais servidores instalados na rede de distribuição. Os clientes locais recebem o fluxo de vídeo gerado diretamente pelo servidor fonte. Por outro lado, os demais clientes recebem fluxos através dos servidores intermediários mais próximos deles.

A representação do *link* entre Brasil e Estados Unidos, bem como os servidores fontes e intermediários disponíveis no serviço de distribuição pode ser visualizado na Figura 87. Nos Estados Unidos foram instalados dois servidores fonte, em Atlanta e em Indianápolis (neste cenário será abordada apenas a transmissão realizada em Atlanta). Todo fluxo gerado por esses servidores foi recebido pelo servidor RJ (PoP de entrada dos fluxos providos do exterior), que por sua vez faz o papel de intermediário para o servidor PB (ou outros servidores, dependendo da configuração do serviço). A Figura 87 ilustra a ligação entre Brasil (*Rio de Janeiro*) e EUA (nós *Atlanta* e *Indianápolis*). Vale ressaltar que essa figura tem objetivo apenas ilustrativo do cenário em questão, pois não é gerada pelo Configurador.

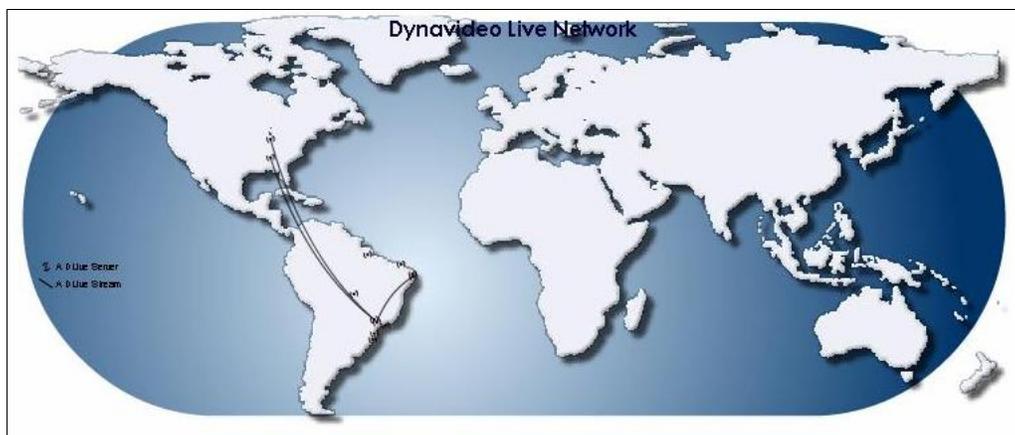


Figura 87 – Representação do *link* Brasil – EUA

6.3 – Considerações Finais

Essa capítulo apresentou dois cenários de uso no qual foram realizados experimentos utilizando a arquitetura de distribuição de vídeo proposta. No Cenário 1 foi apresentado um experimento nacional (SBRC, realizado no Rio Grande do Sul), com transmissão ao vivo). Posteriormente no Cenário 2 foi apresentado um experimento internacional, envolvendo transmissão ao vivo entre Brasil e Estados Unidos (Atlanta/Rio de Janeiro).

Capítulo 7

Considerações Finais

A configuração e a otimização de redes de distribuição são primordiais para o sucesso de aplicações e serviços que envolvam distribuição e transmissão de dados multimídia. Neste sentido, a concepção de estratégias para distribuição eficiente de dados é uma tendência fortemente verificada em aplicações multimídia. Tais estratégias devem identificar as rotas de distribuição e considerar aspectos como proximidade e capacidade de transmissão das redes.

Neste contexto, a infra-estrutura proposta para configuração e gerenciamento do serviço de distribuição de vídeo da RNP promove um melhor atendimento aos clientes, otimizando o tráfego na rede e reduzindo atrasos na distribuição. A infra-estrutura proposta contempla a definição, descrição e identificação das informações como: nós da rede, blocos de endereços, conexões existentes, taxas de transmissão, localização física, bem como servidores e serviços disponíveis. Além disso, a infra-estrutura proposta também contempla uma aplicação gráfica para configuração e gerenciamento dessas informações relativas à rede de distribuição de vídeo digital.

A estratégia implementada tem por base um documento XML contendo a descrição das informações da rede de distribuição. Para tanto foi desenvolvido também um esquema XML (*XML Schema*) que define abstratamente o vocabulário e restrições para os elementos que compõem o cenário que descreve o documento de configuração. A aplicação gráfica que permite a configuração e o gerenciamento do documento de configuração foi desenvolvida usando a tecnologia Java favorecendo a sua portabilidade com diversas plataformas. Assim, está estratégia representa uma boa solução para flexibilizar o uso do serviço *DynaVideo*, enquanto que o documento de configuração descreve de forma clara e objetiva os aspectos da proposta ora apresentada, utilizando tecnologias atuais e passíveis de utilizações futuras.

O serviço de distribuição de vídeo digital já se encontra em funcionamento na rede de distribuição de vídeo digital da RNP, atendendo a diversas entidades espalhadas pelo Brasil e pelo mundo. Pode-se dizer que a arquitetura *X-Conf* proporciona benefícios para o serviço de distribuição de vídeo, tais como, escolha dos melhores recursos disponíveis na rede de distribuição para o atendimento dos clientes, facilidade

na configuração e gerenciamento do serviço de distribuição de vídeo digital, visualização de servidores operantes e inoperantes de forma, bem como nós existentes na rede de distribuição, flexibilidade na manipulação de arquivos textuais (XML), entre outros.

Este trabalho apresenta algumas limitações que podem também ser apontadas como trabalhos futuros. Uma importante limitação da proposta ora apresentada é que o coordenador, ao escolher os melhores recursos disponíveis na rede de distribuição, leva apenas em consideração a capacidade nominal dos enlaces, não considerando o nível de congestionamento dos mesmos. A partir do momento em que a escolha por esses recursos levar em consideração o nível de congestionamento dos enlaces, haverá uma distribuição de recursos de forma mais otimizada e eficiente.

O coordenador ao definir qual servidor irá atender uma determinada requisição, leva em consideração apenas o seu estado (*ativo/inativo*). Este por sua vez não mede o nível da carga dos servidores escolhidos ao atendimento de uma requisição, podendo este ser sobrecarregado no momento em que um número elevado de clientes passam a receber fluxo diretamente do mesmo servidor. Dessa forma, seria importante haver uma melhor distribuição dessas requisições entre os servidores existentes na rede de distribuição.

O desenvolvimento de um protocolo permitiria o coordenador sinalizar ao configurador a atividade ou inatividade de um servidor disponível no documento de configuração. Outra opção seria o desenvolvimento de uma funcionalidade que não permitisse a escrita das informações que descrevem os estados dos servidores (*ativo/inativo*) pelo configurador, deixando esta funcionalidade exclusivamente para o coordenador.

Buscando aprimorar a visualização das informações contidas no documento de configuração, o configurador dispõe da funcionalidade de zoom sobre o cenário utilizado. Para aumentar a flexibilidade na manipulação e visualização dessas informações, a funcionalidade de zoom poderia ser realizada apenas em uma área delimitada pelo administrador.

No momento o *X-Conf* não possui hierarquias de configuração do sistema de distribuição de vídeo digital. O administrador tem total acesso aos dados (leitura/escrita). Políticas hierárquicas que limitem o acesso às informações podem evitar o uso indevido ou mesmo a paralisação de todo serviço.

Referências Bibliográficas

- [1] ALTOVA XML SPY 2005. *XML Editor*. <http://www.altova.com/products_ide.html>. Último acesso em Julho 2005.
- [2] AONDEFICA.COM. *Latitude e Longitude*. <http://www.aondefica.com/lat_3_.asp/>. Último acesso em Julho de 2005.
- [3] APACHE. *Xerces2: Java Parser Readme*. <<http://xml.apache.org/xerces2-j/>>. Último acesso em Agosto 2004.
- [4] BATISTA, C. E. C. F.; SALMITO, T. L.; LEITE, L. E. C.; LEMOS, G.; ELIAS, G. *Big Videos on Small Networks: A Hierarchical and Distributed Architecture for a Video on Demand Distribution Service*. Proceedings of the 1ST IEEE International Conference on Multimedia Services Access Networks (MSAN 2005). June 2005.
- [5] BOAVENTURA, N. P.; *Grafos: Teoria, Modelos, Algoritmos*. Edgar Blücher, 1996.
- [6] BOSCHART, M. A.; KOSA, M. J. *Growing a GUI from XML Tree*. Proceedings of the 8TH Annual Conference on Innovation and Technology in Computer Science Education (ITiCSE'03). June 30 - July 2, 2003.
- [7] BOOCH, G.; RUMBAUGH, J.; JACOBSON, I.; *UML: Guia do Usuário*. Tradução: Fábio Freitas da Silva, Rio de Janeiro, Campus, 2000.
- [8] CAMPBELL, C.; EISENBERG, A. *XML SCHEMA*. ACM SIGMOD Record, Volume 32. Issue 2. Pages 96-101. June 2003.
- [9] COSTELLO, R. L.; *Tutorial XML Schema* <<http://www.xfront.com/xml-schema.html> > Último acesso em abril 2005.
- [10] DAVIDSON, A.; FUCHS, M.; HEDIN, M.; JAIN, M.; MOLONEY, M.; LLOYD, C. *W3C – Schema for Object-Oriented XML 2.0* <<http://www.w3.org/XML/Schema>> Último acesso em Novembro de 2004.
- [11] DEITEL, H. M.; DEITEL, P. J.; *Java, como Programar*. 4^a ed. Porto Alegre: Bookman, 2003.
- [12] GTVD – *Grupo de Trabalho de Vídeo Digital*. Rede Nacional de Ensino e Pesquisa (RNP). <<http://girafa.natalnet.br/gtvd/index.jsp>>. Último acesso em Junho 2005.
- [13] GTVD – *Infra-estrutura da Rede de Distribuição de vídeo Digital*. Rede Nacional de Ensino e Pesquisa (RNP). < <http://girafa.natalnet.br/gtvd/infraestrutura.jsp>> Último acesso em Junho 2005.
- [14] W3CSCHOOLS. *XML Schema Tutorial*. <<http://www.w3schools.com/>>. Último acesso em July 2004.
- [15] W3CSCHOOLS. *XML Tutorial*. <<http://www.w3schools.com/>>. Último acesso em July 2004.
- [16] HAROLD, E. R.; *Processing XML with Java: A Guide to SAX, DOM, JDOM, JAXP and TrAX*. <<http://www.cafeconleche.org/books/xmljava/>>. Último acesso em Outubro 2005.

- [17] HARREN, M; RAGHAVACHARI, M; SHMUELE, O; BURKE, M; SARKAR, V; BORDAWEKAR, R; *XJ: Integration of XML Processing into Java*. Proceedings of the 13th International World Wide Web Conference. May 2004.
- [18] IBM. *Understanding DOM. Presented by developersWorks, your source for great tutorials*.
<<http://www.digilife.be/quickreferences/PT/Understanding%20DOM.pdf>> Último acesso em Setembro 2003.
- [19] KAIVA, H; FURNT, B; *Techniques for Improving the Capacity of Video-on-Demand Systems*. Proceedings of the 29th Annual Hawaii International Conference on System Sciences (HICSS-29). 1996.
- [20] LEE, S; YOUNG, K; MOON, Y; SONG, Y; *Dynamic Buffer Allocation in video-on-Demand Systems*. Proceedings of the 2001 ACM SIGMOD International Conference on Management of Data. Pages 343-354. 2001.
- [21] MA, H; SHIN, K, G. *Multicast Video-on-Demand Services*, ACM SIGCOMM Computer Communication Review. Volume 32. Issue 1. Pages 31-43. January 2002.
- [22] MAHANTI, A; EAGER, D; VERNON, M, K; STUKEL, D, S; *Scalable On-Demand Media Streaming with Packet Loss Recovery*. ACM SIGCOMM – A Special Interest Group on Data Communications, 2001.
- [23] MIYAZAKI, Y; NAHRSTEDT, K; *Dynamic Coordination of Movie According to Popularity Index and Resources Availability within a Hierarchical VoD System*. Proceedings of the IEEE TENCON - Speech and Image Technologies for Computing and Telecommunications, 1997.
- [24] POCHUEVA, J; MUNSON, E; POCHUEVA, D; *Optimizing Video-On-Demand through Requestcasting*. Proceedings of the 7st ACM International Conference on Multimedia (ACM Multimedia 1999). November 1999.
- [25] PINHO, L, B; ISHIKAWA, E; *GLOVE – A Distributed Environment for Scalable Video-on-Demand Systems*. The International Journal of High Performance Computing Applications. Volume 17. Pages 147-161. 2003.
- [26] RFC 1519. *Classless Inter-Domain Routing (CIDR): An Address Assignment and Aggregation Strategy*. <<http://www.faqs.org/rfcs/rfc1519.html>>. Julho 2002.
- [27] RNP – REDE NACIONAL DE ENSINO E PESQUISA. *Operação de Backbone*. <<http://www.rnp.br/backbone/index.php>>. Último acesso em Julho 2005.
- [28] SALMITO, T.; FARIAS, J. P, ELIAS, G.; LEMOS G; LEITE, L. *Uma Arquitetura Hierárquica e Distribuída para um Serviço de Distribuição de Vídeo sob Demanda*. 10^o Simpósio Brasileiro de Sistemas Multimídia e Web (WebMedia 2004). Ribeirão Preto-SP. 2004.
- [29] SHEU, JP; WANG, HL; CHANG, CH; TSENG, YC; *A Fast Video-on-Demand Broadcasting Scheme for Popular videos*. IEEE Transaction on Broadcasting, Vol. 50, No. 2, June 2004, pp. 120-125.
- [30] SHI, W; GHANDEHARIZADEH, S; *Buffer Sharing in Video-On-Demand Servers*. ACM SIGMETRICS Performance Evaluation Review. Special Issue on Multimedia Storage Systems. Volume 25. Issue 2. Pages 13-20. September 1997.

- [31] SILVA, L. O.; *Serviço de Coordenação de Servidores de Vídeo*. Trabalho de Conclusão de Curso (Bacharelado em Ciência da Computação). Departamento de Informática e Matemática Aplicada (DIMAp). Universidade Federal do Rio Grande do Norte (UFRN). Natal – RN. 2003.
- [32] SUZUKI, J; YAMAMOTO, Y.; *Managing the Software Design Documents with XML*. Proceedings of the 16th Annual International Conference on Computer Documentation. Pages 127-136. July 1998.
- [33] SZWARCFITER, J. L.; *Grafos e Algoritmos Computacionais*. Editora Campus, 1986.
- [34] UFMG – *Projetos de Video sob Demanda*.
<<http://www.vod.dcc.ufmg.br/vod/docs/descricao/dccvod.html>>. Último acesso em Julho 2005.
- [35] TANEMBAUM, A. S.; *Redes de Computadores*. 12^a ed. Rio de Janeiro: Campus, 1997.
- [36] THE JAVA WEB SERVICES TUTORIAL. *Validating with XML Schema*.
<<http://java.sun.com/webservices/docs/1.0/tutorial/doc/JAXPDOM9.html>> Ultimo acesso em Agosto de 2004.
- [37] VOHRA, D.; *Document Validation with a XML Schema*.
<<http://www.onjava.com/pub/a/onjava/2004/09/15/schema-validation.html>>
Ultimo acesso em Outubro de 2004.
- [38] XML – EXTENSIBLE MARKUP LANGUAGE.
<http://www.gta.ufrj.br/grad/00_1/miguel/index.html> Ultimo acesso em Dezembro de 2004.
- [39] 3COM. *Understanding IP Addressing: Everything You Ever Wanted To Know*.
<http://www.3com.com/other/pdfs/infra/corpinfo/en_US/501302.pdf>. Último acesso em Julho 2005.
- [40] COMEN, T. H; LEISERSON, C. E; RIVEST, R. L; STEIN, C. *Introduction to Algorithms*, Second Edition, MIT Press, 2001.

Apêndice I

Noções Fundamentais de XML

A base da arquitetura proposta se dá através de um documento de configuração baseado na linguagem XML e *XML Schema*. Visando um melhor entendimento pelo leitor do uso dessa linguagem no escopo deste do trabalho, serão abordados alguns conceitos básicos tanto linguagem XML bem como *XML Schema*.

I.1 – O que é XML?

XML (*Extensible Markup Language*) é uma linguagem de marcação extensível (*meta-markup language*) porque não pré define nenhuma *tag*, provê um formato para descrever dados estruturados. Diferentemente da maioria das linguagens de programação, inclusive a própria HTML (*HyperText Markup Language* – Linguagem de marcação de hipertexto), não apresenta um conjunto limitado de *tags* a serem utilizadas. XML permite a declarações de *tags* mais precisas objetivando resultados mais significativos de busca através de múltiplas plataformas. É uma linguagem bastante flexível, pois permite aos seus usuários definirem suas próprias *tags* infinitamente. A criação de *tags* se dá através de dados estruturados.

Para a definição de um conjunto de *tags*, XML oferece uma estrutura padrão, possibilitando ao usuário a criação de sua própria estrutura ou então usar estruturas já definidas anteriormente. A grande vantagem de utilizar XML na criação de novas linguagens é o fato que todas linguagens baseadas em XML, compartilham uma sintaxe básica comum, como por exemplo *XML Schema*.

A linguagem XML foi desenvolvida pela SGML (*Standard Generalized Markup Language*), tornando-se um padrão especificado pela ISO 8879 em 1986. XML surgiu com intuito de ser uma linguagem leve, simples e que pudesse ser utilizada na *web*, possibilitando uma correspondente habilidade em manipular e procurar por dados através de aplicações remotas. Além de ser utilizada para representação de dados, também é muito utilizada na troca de informações entre diferentes aplicações, tornando assim um elo comum de entendimento entre elas.

I.2 – Noções de HTML

Atualmente muitas páginas da Internet são desenvolvidas utilizando a linguagem HTML. O termo *hypertext* é definido por textos que têm *links* para outros textos. Já o termo *markup language* define anotações para a estrutura de um texto [38].

Um documento HTML possui duas características importantes:

1. São desenvolvidos para prover estrutura lógica da informação destinada à apresentação de páginas da rede mundial de computadores.
2. Contém um conjunto de *tags* limitadas destinadas a definir a estrutura do documento. Cada *tag* possui a sua semântica pré-definida na qual são especificamente definidas para páginas *web*. Abaixo segue um breve comparativo entre XML e HTML para que haja melhor compreensão do uso do XML no escopo deste trabalho.

I.3 – Comparação entre HTML e XML

Muitas bibliografias fazem referencia ao HTML como sendo primo do XML, pois além serem desenvolvidas pelo SGML, utiliza sintaxes similares. A grande diferença entre documentos HTML e documentos XML é que o HTML descreve a aparência e a ações em uma página *web*, enquanto o XML não descreve aparência e ações, mas sim o que cada trecho de dados é ou representa. Em outras palavras, o XML descreve o conteúdo propriamente dito do documento.

Semelhante ao HTML, o XML também faz uso de *tags* (palavras encapsuladas por sinais '<' e '>') e atributos (definidos com *name="value"*). No entanto, o HTML especifica o sentido para as *tags* e atributos (e freqüentemente a maneira pela qual o texto entre eles será exibido em um navegador), enquanto XML usa as *tags* somente para delimitar trechos de dados, e deixa a interpretação do dado a ser realizada completamente para a aplicação que o está lendo. Por exemplo, uma *tag* <p> em um documento HTML indica um parágrafo, enquanto no XML essa *tag* pode indicar um preço, um parâmetro, uma pessoa, ou qualquer outra coisa que se possa imaginar (inclusive algo que não tenha nada a ver com a letra “p” como, por exemplo, servidores de vídeo).

As regras de formatação para documentos XML são mais rígidas do que para documentos HTML. Uma *tag* esquecida ou um atributo sem aspas, por exemplo, torna o documento inutilizável, enquanto que no HTML isso pode ser tolerado.

I.4 – Características da linguagem XML

A linguagem XML provê uma representação estruturada dos dados, mostrando ser amplamente utilizada no desenvolvimento e uso de aplicações. O XML é otimizado para distribuição através da *web*, sendo este definido pelo *W3C (Word Wide Web Consortium)*, assegurando que os dados estruturados serão uniformes e independentes de aplicações e fornecedores [38].

O XML provê um padrão que pode codificar o conteúdo, as semânticas e os esquemas para uma grande variedade de aplicações. Dentre elas se destacam:

- Um simples documento.
- Um registro estruturado tal como uma ordem dos servidores de vídeo.
- Um objeto com métodos e dados como objetos Java.
- Um registro de dados. (ex. resultado de uma consulta a bancos de dados).
- Apresentação gráfica, como interface de aplicações de usuário.
- Entidades e tipos de esquema padrões.

O XML é considerado de grande importância na Internet e *intranets* porque provê a capacidade de interoperabilidade entre diferentes sistemas. Possui um padrão aberto, flexível e independente de dispositivo.

As principais funções de um documento XML são:

- Descrever dados;
- Apresentar dados em algum formato como HTML;
- Transportar dados;
- Trocar dados de forma transparentes entre diferentes plataformas.

I.5 – Principais benefícios da linguagem XML

A linguagem XML tem como principal objetivo trazer flexibilidade e desempenho às aplicações voltadas principalmente para *Web*. Dentre os benefícios destacam-se:

- Buscas eficientes, devido os dados ser unicamente etiquetados, o que permite, por exemplo, buscas mais rápidas sem a necessidade da aplicação de busca saber detalhes da esquematização ou construção do banco de dados.
- Desenvolvimento de aplicações (*Web*) mais flexíveis, incluindo a integração dos dados de diferentes fontes;
- Computação e manipulação dos dados localmente.
- Utilizar ferramentas como o XML DOM para realizar a manipulação dos dados contidos no documento XML. Podendo ocorrer via *scripts* ou através de uma linguagem de programação;
- Múltiplas formas de visualização através dos recursos CSS e XSL;
- Distribuição dos dados via rede de forma mais comprimida – devido seu formato texto, pode ser distribuído via http;
- Adoção de padrões abertos;
- Escalabilidade por separarem completamente os dados da parte lógica da forma que será visualizada;

I.6 – Separação entre Dados e Apresentação

A mais importante característica do XML se resume em separar a interface com o usuário (apresentação) dos dados estruturados (lógica). O HTML especifica como o documento deverá ser apresentado na tela por meio de um navegador qualquer. Já o XML define o conteúdo propriamente dito do documento. Por exemplo, em HTML são utilizadas *tags* para definir tamanho e cor de fonte, assim como formatação de parágrafo. No XML utilizam-se as *tags* para descrever os dados, como exemplo *tags* endereços IP, pontos de rede, servidores ativos, conexões, etc.

O XML ainda conta com recursos tais como folhas de estilo definidas com XSL (*Extensible Style Language*) e CSS (*Cascading Style Sheets*) para a apresentação de dados em um navegador. Essa separação dos dados da apresentação permite a integração dos dados de diversas fontes. Estas informações podem ser convertidas para XML permitindo que os dados sejam trocados via rede tão facilmente como são realizadas com páginas HTML.

I.2 – Definição conceitual do XML

Nessa seção descreve alguns conceitos da linguagem XML, bem como a linguagem HTML.

I.2.1 – Estrutura do Documento

Um documento XML é uma árvore rotulada como ilustra a figura abaixo, consiste de vários elementos externos e internos. Os elementos externos de um documento XML são rotulados como segue abaixo:

- Uma seqüência de texto (Documento como um todo);
- Cabeçalho do documento (informações para os processadores);
- Comentários;
- Documentos de validação, *XML SCHEMA* ou DTD (*Document Type Declaration*);

Além das informações externas, um documento XML possui os elementos internos, sendo estes as *tags* propriamente ditas. Esses elementos podem ser rotulados como:

- *Tags* – um documento XML é um arquivo texto com *tags* de marcação que contém informações para outras aplicações; são delimitadas pelos sinais de “>” (maior) e “<” (menor) respectivamente.
- *Elementos* – Estão posicionados entre as delimitações das *tags* contendo informações referentes a algo que se deseja representar no documento como um todo. Um elemento é dito vazio quando não apresenta elementos ou atributos.
- *Atributos* – os atributos representam uma ou mais características de um elemento.
- *Nomes de elementos e Atributos* – são os identificadores desses elementos ou atributos, geralmente utilizam-se nomes dos quais estão sendo representados, facilitando assim, as possíveis correções que venham ocorrer.
- *Valor dos atributos* – é definido pelo conteúdo ou valor assumido por eles.

A Figura 1 ilustra a representação de uma árvore em XML

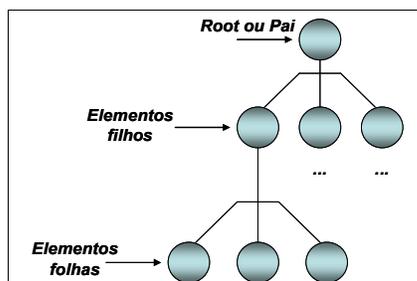


Figura I.1 – Representação de uma árvore XML

A Figura 2 ilustra o formato das *tags*, dos elementos e atributos respectivamente.

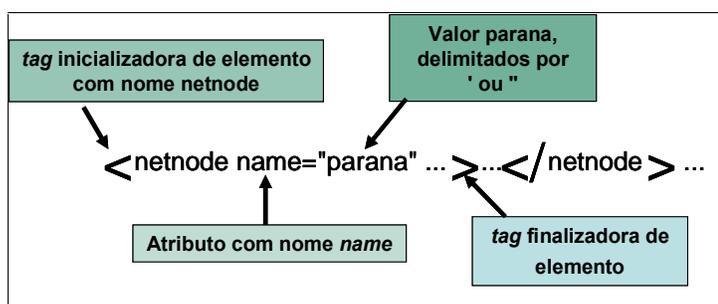


Figura I.2 – Representação de uma *tag*, elementos e atributos XML

Como mencionado anteriormente, um elemento vazio é aquele elemento que não possui elementos ou mesmo atributos. A Figura 3 ilustra um elemento vazio.

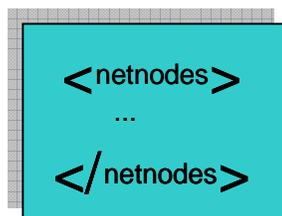


Figura I.3 – Representação de um elemento vazio em XML

Um documento XML é dito bem formatado quando segue algumas regras básicas. Tais regras são mais simples do que para documentos HTML e permitem que os dados sejam lidos e expostos sem nenhuma descrição externa como, por exemplo, sua validação contra um *Schema*. Faz distinção entre letras maiúsculas e minúsculas e permite ao usuário adicionar comentários no código XML através da *tag* `<!-- comentário -->`, sendo este ignorado por todos os processadores. Um documento pode ser bem estruturado quando:

- Possui casamento das *tags* de início e fim;

- As *tags* de elementos devem ser posicionadas de forma correta e apropriada;
- Não pode haver sobreposição de elementos, por exemplo:

```
<title>Distribuição de Vídeo Digital
<formato>mpeg2
</title>
Fernando Luiz de Almeida
</formato>
```

- Os elementos devem aparecer seguindo uma ordem, sem sobreposição. O exemplo abaixo irá ilustrar a forma correta das *tags* citadas acima.

```
<title>Distribuição de Vídeo Digital
  <formato>mpeg2</formato>
  <autor>Fernando Luiz de Almeida</autor>
</title>
```

- Caracteres especiais devem ser analisados antes do uso, só poderão ser utilizados se fizerem referência a caracteres *Unicode*¹².
 - Exemplo: `&=&`

I.3 – Validação de Documentos XML

Um documento XML pode conter regras que definem a estrutura do documento. Por sua vez, estas regras são estabelecidas por DTDs (*Document Type Definitions*) ou *Schemas*.

Os documentos *DTDs* e *Schemas* são responsáveis em validar as informações contidas em um documento XML. Nos dias atuais, o *Schema* XML vem sendo amplamente utilizado devido sua maior flexibilidade, semelhança em sua sintaxe além de utilizar a própria linguagem XML em sua definição. A seguir será mencionado alguns conceitos básicos sobre *DTDs* e *Schemas*, sendo dado maior ênfase no XML *Schema*, utilizado para validar o documento XML deste trabalho.

I.3.1 – DTD (*Document Type Definition*)

DTDs são documentos utilizados para validar um documento XML. Um documento é dito válido se estiver de acordo com o esquema definido, seja ele um *DTD*

¹² *Unicode* – padrão que fornece um único número para cada caracter, não importando a plataforma, o programa, ou mesmo idioma.

ou um XML *Schema*. Um analisador de documentos pode checar os dados analisando as regras contidas no *DTD* para ter certeza de que o dado foi estruturado corretamente. Os dados enviados sem *DTD* são conhecidos como dados bem formatados.

Atribui-se a um *DTD* a responsabilidade de usar uma linguagem oficial para delimitar a estrutura e os possíveis valores que os documentos XML podem assumir. Basicamente, um *DTD* define um vocabulário comum entre documentos *xml*.

Um documento válido pode ser dito que está bem formatado, mas um documento bem formatado pode não estar validado. Isso ocorre devido ao documento estar bem-formatado e não estar de acordo com o esquema definido. Os *DTDs* são formas de se descrever classes de documentos XML (como gramáticas para outras linguagens). Os principais benefícios do uso de *DTDs* são:

- Impor uma estrutura conforme definida pelo *DTD*;
- A aplicação acessa a estrutura do documento;
- *DTD* permite a declaração de valores padrões ou fixos para os atributos, resultando em um documento menor.

As principais restrições do uso de *DTDs*:

- A linguagem dos *DTDs* possui uma sintaxe diferente do XML;
- Sintaxe complexa e de pouco entendimento;
- Inconsistente devido suas limitações;
- Capacidade limitada na especificação de tipo de dados;
- Possui apenas 10 tipos de dados definidos;

Na Figura 4 segue um exemplo de *DTD*:

```
<!DOCTYPE videocollection [  
<!ELEMENT video  
(title,author?,date?,description,ingredients,preparation,related)>  
<!ATTLIST video id ID  
#REQUIRED  
category (breakfast|lunch|dinner|dessert|unknown)  
#IMPLIED>  
<!ELEMENT title (#PCDATA)>  
<!ELEMENT author ANY>  
...  
>
```

Figura I.4 – Representação de um modelo de *DTD*

I.3.2 – XML *Schema*

Devido as limitações e problemas existentes nos *DTDs*, fez com que a W3C desenvolvesse outras linguagens de esquemas (*Schema languages*) tais como DSD,

XML *Schema*, etc [18]. XML *Schema* possui uma estrutura que define as regras, restrições dos elementos e atributos de um documento XML. Define o formato dos elementos, atributos e possíveis valores que estes possam receber. Um documento XML que estiver de acordo com o *Schema* é dito documento válido. As principais características de um documento XML *Schema* são:

- ✓ Linguagem bastante flexível devido ao uso da própria linguagem XML na formação de *tags*;
- ✓ Possui um conjunto de tipos que são compatíveis com a maioria dos bancos de dados atuais;
- ✓ Disponibiliza elementos que impõem regras e restrições a outros elementos;
- ✓ Definição de 44 tipos de dados (*string*, *date*, *integer*, *decimal*, *boolean*, *timeDurations*, etc);
- ✓ Permite a união e definição de novos tipos, derivação de tipos por extensão e por restrição, além de controlar tipos enumerados;
- ✓ Alto nível de confiabilidade e flexibilidade.

Existem dois tipos básicos de elementos encontrados nos documentos XML *Schema*: elementos simples e elementos complexos. Um elemento simples pode ser definido com a utilização ao elemento *simpleType*, normalmente utilizado quando se deseja impor alguma restrição ao elemento. Um elemento declarado como *simpleType*, não possui atributos. As declarações de tipos complexos exigem a utilização do elemento *complexType*. Elementos declarados como *complexType* são aqueles que possuem outros elementos ou atributos. As Figuras 5 e 6 ilustram uma declaração do tipo *simpleType* e *complexType* respectivamente.

```
<xs:attribute name="default" use="optional">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:pattern value="yes|no"/>
    </xs:restriction>
  </xs:simpleType>
</xs:attribute>
```

Figura I.5 – Representação de um elemento *simpleType*

```
<xs:element name="to" maxOccurs="unbounded">
  <xs:complexType>
    <xs:attribute name="name" use="required">
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:pattern value="([a-z])+"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
  </xs:complexType>
</xs:element>
```

```
</xs:restriction>
</xs:simpleType>
</xs:attribute>
...
</xs:complexType>
</xs:element>
```

Figura I.6 – Representação de um elemento *complexType*

A linguagem XML *Schema* disponibiliza dois atributos para a definição de cardinalidade na declaração de elementos: *minOccurs* e *maxOccurs*. O atributo *minOccurs* é o elemento responsável por determinar o número mínimo em que um elemento deve aparecer. Quando omitido da declaração assume-se este elemento aparecerá uma vez. Já o elemento *maxOccur* determina a quantidade máxima de vezes em que um elemento poderá aparecer. Caso este elemento também seja omitido do documento, tem-se a opção *unbounded* que permite um número infinito de vezes ou então por *default* aparecer uma vez.

O XML *Schema* define três delimitadores de grupos para a definição da ordem em que os elementos devem aparecer dentro do documento XML válidos pelo esquema. O delimitador *sequence* informa que os sub-elementos devem aparecer no documento XML na mesma ordem em que se apresentam no documento XML *Schema*. O delimitador *choice* informa apenas um dos sub-elementos declarados no *Schema* deve aparecer no documento. E pra finalizar, o delimitador *all* fica responsável em informar que todos os elementos declarados no *Schema* podem aparecer uma ou nenhuma vez no documento.

Existem duas formas de se especificar um XML *Schema* para uma determinada aplicação. Este pode ser referenciado dentro do próprio documento através de *namespaces* ou então por aplicações externas, por exemplo, *Xerces Java*, *JAXP*, *JAXB*, etc.

- **Referência interna:** A referência a um XML *Schema* pode ser feita dentro do próprio documento XML, fazendo uso de atributos e *namespaces*. O código ilustrado na Figura 7 mostra um exemplo de *namespace* fazendo a referencia a um XML *Schema*. O esquema definido aqui é denominado *manager.xsd*.

```
<?xml version="1.0"?>
<gtvideo xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
xsi:NamespaceSchemaLocation="C://Fernando/manager.xsd">
...
```

Figura I.7 – Representação do uso de *namespace*

Figura I.9 – Representação de um documento XML

I.3.4 – Definição de elementos simples e tipos de dados

Pode se dizer que um documento XML é basicamente constituído por elementos e conseqüentemente, atributos desses elementos. O maior problema na definição de um elemento no XML *Schema* é a criação de nomes e a associação desse nome com um tipo (*string*, *integer*, *data*, *etc*). A Figura 10 ilustra os elementos associados a um determinado tipo.

```
<xsd:element name="date" type="xsd:date"/>
<xsd:element name="name" type="xsd:string"/>
<xsd:element name="address" type="xsd:integer"/>
<xsd:element name="description" type="xsd:string" />
<xsd:element name="place" type="xsd:string" />
```

Figura I.10 – Elementos associados a um tipo pré-definido

O elemento *date* possui um formato pré-definido (*yyyy.mm.dd*), sendo ano, mês e dia. Esta definição é padronizada pelo *W3C XML Schema Recommendation*.

I.3.5 – Restrição de valores

Além da criação de elementos, o usuário pode criar novos tipos simples de dados. Esses tipos descrevem uma faixa de valores numéricos, formato de números telefônicos, celulares, número de produtos, uma lista enumerada, entre outras, ou seja, o valor máximo e mínimo que poderá ser assumido pelo documento XML. A Figura 11 ilustra a definição de um elemento simples com restrição numérica de valores.

```
<xsd:simpleType name="idNumber">
  <xsd:restriction base="xsd:integer">
    <xsd:minInclusive value="1" />
    <xsd:maxInclusive value="100000" />
  </xsd:restriction>
</xsd:simpleType>
```

Figura I.11 – Representação de restrições de valores

O elemento *idNumber*, possui um tipo de dados definido como *integer*, além da restrição imposta, permitindo receber valores na faixa entre 1 e 100000.

I.3.6 – Enumeration

Existe uma outra forma de restringir valores aos elementos e atributos através do delimitador *Enumeration*. Este por sua vez permite ao usuário criar uma lista enumerada, pré-

estabelecendo os valores que um elemento poderá receber. A Figura 12 ilustra o uso do delimitador *Enumeration*.

```
<xs:simpleType>  
  <xs:restriction base="xs:string">  
    <xs:enumeration value="bps"/>  
    <xs:enumeration value="Kbps"/>  
    <xs:enumeration value="Mbps"/>  
    <xs:enumeration value="Gbps"/>  
  </xs:restriction>  
</xs:simpleType>
```

Figura I.12 – Delimitador de restrição *Enumeration*

I.4 – Uma noção sobre DOM

DOM é uma API (*Applications Programming Interface*) independente de plataforma e linguagem de programação usada para manipular documentos XML. É através do XML DOM que o usuário poderá navegar pela estrutura do documento XML, adicionar novos elementos e atributos, bem como modificações, remoções e consultas.

Para realizar a manipulação do documento XML, este deverá ser carregado em memória, este método é denominado *parser*¹³. Após carregar o documento XML em memória, o DOM monta esse documento em forma de árvores, método utilizado para realizar a busca por informações. A Figura 13 ilustra parte de um documento XML após a realização do *parser* em memória pelo DOM.

¹³ Método utilizado para carregar o documento XML em memória para que possa ser lido e manipulado pela aplicação.

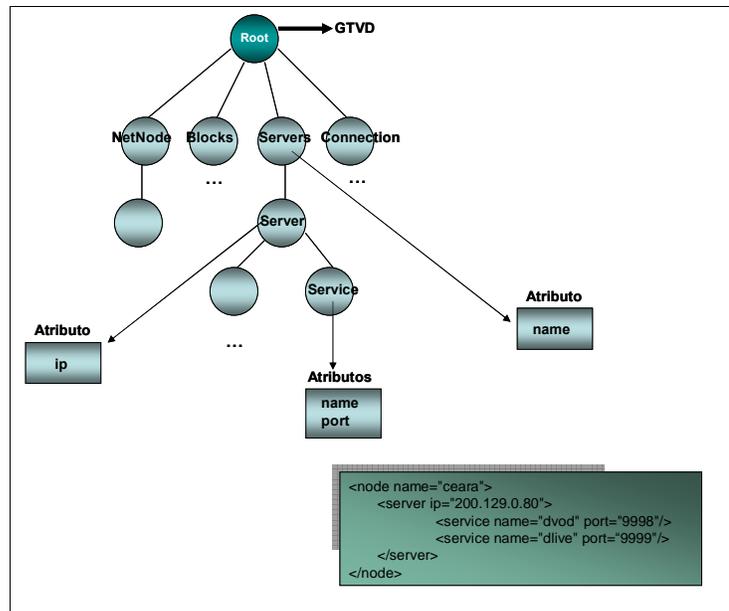


Figura I.13 – Estrutura de árvore com o uso do DOM

A principal desvantagem do DOM ocorre quando um documento XML for extenso, pois este carrega todo documento XML em memória antes do seu uso. Em contrapartida, uma vez carregado em memória, à busca por informações se torna bastante rápida. Existem quatro níveis de implementação usando o DOM, sendo a de nível 3 utilizada neste trabalho:

- **Nível 0:** Funções existentes conhecidas das linguagens *script* dos *browsers*.
- **Nível 1:** Funcionalidade para navegação em documentos e manipulações.
- **Nível 2:** Adiciona modelos de *style sheets* (folhas de estilo), filtros, modelos de eventos, e suporte a *namespaces*.
- **Nível 3:** Nível utilizado na implementação deste trabalho, pois permite carregar, manipular e salvar documentos, bem como o uso de *DTDs* e *Schemas* tornando válido um documento XML, entre outras características.

Apêndice II

Este Apêndice descreve o documento de configuração completo, que está em funcionamento no sistema de distribuição de vídeo digital da rede RNP.

Documento de Configuração (Arquivo XML)

```
<netdescr xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="C:\Documents and Settings\Fernando\My
Documents\managerQuali.xsd">
  <netnodes>
    <node name="Fio Cruz" local="Rio de Janeiro"
latitude="-24.33377778" longitude="-45.2075" default="no" hasServer="yes" />
    <node name="Rio de Janeiro" local="Rio de Janeiro"
latitude="-22.90277778" longitude="-43.2075" default="yes" hasServer="yes" />
    <node name="Parana" local="Curitiba" latitude="-
25.42777778" longitude="-49.27305556" default="no" hasServer="no" />
    <node name="Acre" local="Rio Branco" latitude="-
9.974722222" longitude="-67.81" default="no" hasServer="no" />
    <node name="Minas Gerais" local="Belo Horizonte"
latitude="-19.92083333" longitude="-43.93777778" default="no" hasServer="no" />
    <node name="Sao Paulo" local="Sao Paulo" latitude="-23.5475" longitude="-
46.63611111" default="no" hasServer="yes" />
    <node name="Pernambuco" local="Recife" latitude="-8.053888889"
longitude="-34.88111111" default="no" hasServer="yes" />
    <node name="Santa Catarina" local="Florianopolis" latitude="-27.59666667"
longitude="-48.54916667" default="no" hasServer="yes" />
    <node name="Paraiba" local="Campina Grande" latitude="-7.230555556"
longitude="-35.88111111" default="no" hasServer="yes" />
    <node name="Distrito Federal" local="Brasilia" latitude="-15.76305556"
longitude="-47.92972222" default="no" hasServer="yes" />
    <node name="Rondonia" local="Porto Velho" latitude="-8.761944444"
longitude="-63.90388889" default="no" hasServer="no" />
    <node name="Amapa" local="Macapa" latitude="0.038888889" longitude="-
51.06638889" default="no" hasServer="no" />
    <node name="Para" local="Belem" latitude="1.455833333" longitude="-
48.50444444" default="no" hasServer="no" />
    <node name="Mato Grosso" local="Cuiaba" latitude="-15.59611111"
longitude="-56.09666667" default="no" hasServer="no" />
    <node name="Tocantins" local="Palmas" latitude="-10.21277778"
longitude="-48.36027778" default="no" hasServer="no" />
    <node name="Amazonas" local="Manaus" latitude="-3.101944444"
longitude="-60.025" default="no" hasServer="no" />
    <node name="Mato Grosso do Sul" local="Campo Grande" latitude="-
20.44277778" longitude="-54.64638889" default="no" hasServer="no" />
    <node name="Goias" local="Goiania" latitude="-16.68" longitude="-
49.25388889" default="no" hasServer="no" />
  </netnodes>
</netdescr>
```

```
<node name="Ceara" local="Fortaleza" latitude="-3.717222222" longitude="-38.543055556" default="no" hasServer="yes" />
<node name="Rio Grande do Norte" local="Natal" latitude="-5.795" longitude="-35.209444444" default="no" hasServer="yes" />
<node name="Piaui" local="Terezina" latitude="-5.089166667" longitude="-42.801944444" default="no" hasServer="no" />
<node name="Maranhao" local="Sao Luis" latitude="-2.529722222" longitude="-44.302777778" default="no" hasServer="no" />
<node name="Espirito Santo" local="Vitoria" latitude="-20.319444444" longitude="-40.337777778" default="no" hasServer="no" />
<node name="Sergipe" local="Aracaju" latitude="-10.911111111" longitude="-37.071666667" default="no" hasServer="no" />
<node name="Alagoas" local="Maceio" latitude="-9.665833333" longitude="-35.735277778" default="no" hasServer="no" />
<node name="Roraima" local="Boa Vista" latitude="2.819722222" longitude="-60.673333333" default="no" hasServer="no" />
<node name="Bahia" local="Salvador" latitude="-12.971111111" longitude="-38.510833333" default="no" hasServer="no" />
<node name="Rio Grande do Sul" local="Porto Alegre" latitude="-30.033055556" longitude="-51.23" default="no" hasServer="yes" />
</netnodes>
```

```
<connections>
```

```
<source name="Parana">
  <target name="Sao Paulo" rate="155" unit="Mbps" />
</source>
<source name="Minas Gerais">
  <target name="Rio de Janeiro" rate="155" unit="Mbps" />
</source>

<source name="Acre">
  <target name="Distrito Federal" rate="4" unit="Mbps" />
</source>
<source name="Fio Cruz">
  <target name="Rio de Janeiro" rate="155" unit="Mbps" />
</source>
```

```
<source name="Rio de Janeiro">
  <target name="Fio cruz" rate="155" unit="Mbps" />
  <target name="Sao Paulo" rate="622" unit="Mbps" />
  <target name="Distrito Federal" rate="155" unit="Mbps" />
  <target name="Minas Gerais" rate="155" unit="Mbps" />
  <target name="Sergipe" rate="34" unit="Mbps" />
  <target name="Pernambuco" rate="34" unit="Mbps" />
  <target name="Ceara" rate="34" unit="Mbps" />
  <target name="Bahia" rate="34" unit="Mbps" />
  <target name="Maranhao" rate="34" unit="Mbps" />
  <target name="Espirito Santo" rate="34" unit="Mbps" />
  <target name="Para" rate="34" unit="Mbps" />
```

```
<target name="Alagoas" rate="34" unit="Mbps" />
<target name="Paraiba" rate="34" unit="Mbps" />
<target name="Piaui" rate="4" unit="Mbps" />
<target name="Amazonas" rate="8" unit="Mbps" />

</source>

<source name="Rio Grande do Sul">
  <target name="Sao Paulo" rate="155" unit="Mbps" />
</source>

<source name="Sao Paulo">
  <target name="Rio de Janeiro" rate="622" unit="Mbps" />
  <target name="Distrito Federal" rate="155" unit="Mbps" />
  <target name="Rio Grande do Sul" rate="155" unit="Mbps" />
  <target name="Santa Catarina" rate="155" unit="Mbps" />
  <target name="Parana" rate="155" unit="Mbps" />
  <target name="Rio Grande do Norte" rate="34" unit="Mbps" />
  <target name="Roraima" rate="4" unit="Mbps" />
  <target name="Amapa" rate="1" unit="Mbps" />
</source>

<source name="Pernambuco">
  <target name="Rio de Janeiro" rate="34" unit="Mbps" />
</source>

<source name="Santa Catarina">
  <target name="Sao Paulo" rate="155" unit="Mbps" />
</source>

<source name="Paraiba">
  <target name="Rio de Janeiro" rate="34" unit="Mbps" />
</source>

<source name="Distrito Federal">
  <target name="Rio de Janeiro" rate="155" unit="Mbps" />
  <target name="Sao Paulo" rate="155" unit="Mbps" />
  <target name="Mato Grosso" rate="34" unit="Mbps" />
  <target name="Mato Grosso do Sul" rate="34" unit="Mbps" />
  <target name="Goias" rate="34" unit="Mbps" />
  <target name="Tocantins" rate="4" unit="Mbps" />
  <target name="Rondonia" rate="4" unit="Mbps" />
  <target name="Acre" rate="4" unit="Mbps" />
</source>

<source name="Rondonia">
  <target name="Distrito Federal" rate="4" unit="Mbps" />
</source>
```

```
<source name="Amapa">
  <target name="Sao Paulo" rate="4" unit="Mbps" />
</source>

<source name="Para">
  <target name="Rio de Janeiro" rate="8" unit="Mbps" />
</source>

<source name="Mato Grosso">
  <target name="Distrito Federal" rate="34" unit="Mbps" />
</source>

<source name="Tocantins">
  <target name="Distrito Federal" rate="4" unit="Mbps" />
</source>

<source name="Amazonas">
  <target name="Rio de Janeiro" rate="8" unit="Mbps" />
</source>

<source name="Mato Grosso do Sul">
  <target name="Distrito Federal" rate="34" unit="Mbps" />
</source>

<source name="Goias">
  <target name="Distrito Federal" rate="34" unit="Mbps" />
</source>

<source name="Piaui">
  <target name="Rio de Janeiro" rate="34" unit="Mbps" />
</source>

<source name="Maranhao">
  <target name="Rio de Janeiro" rate="34" unit="Mbps" />
</source>

<source name="Ceara">
  <target name="Rio de Janeiro" rate="34" unit="Mbps" />
</source>

<source name="Rio Grande do Norte">
  <target name="Sao Paulo" rate="34" unit="Mbps" />
</source>

<source name="Espirito Santo">
  <target name="Rio de Janeiro" rate="34" unit="Mbps" />
</source>

<source name="Bahia">
  <target name="Rio de Janeiro" rate="34" unit="Mbps" />
```

```
</source>

<source name="Sergipe">
  <target name="Rio de Janeiro" rate="34" unit="Mbps" />
</source>

<source name="Alagoas">
  <target name="Rio de Janeiro" rate="34" unit="Mbps" />
</source>

<source name="Roraima">
  <target name="Sao Paulo" rate="4" unit="Mbps" />
</source>
</connections>

<blocks>
  <range name="Parana">
    <block address="182.255.249.255" mask="24" />
    <block address="200.17.251.0" mask="24" />
    <block address="200.134.0.0" mask="16" />
    <block address="200.238.190.0" mask="24" />
    <block address="200.19.64.0" mask="20" />
    <block address="200.17.96.0" mask="20" />
    <block address="200.17.192.0" mask="20" />
    <block address="200.17.208.0" mask="20" />
    <block address="200.17.224.0" mask="20" />
    <block address="200.17.240.0" mask="20" />
    <block address="200.238.159.0" mask="24" />
  </range>

  <range name="Acre">
    <block address="200.129.173.0" mask="24" />
    <block address="200.129.174.0" mask="24" />
    <block address="200.129.175.0" mask="24" />
    <block address="200.129.181.0" mask="24" />
  </range>

  <range name="Fio Cruz">
    <block address="157.86.0.0" mask="16" />
  </range>

  <range name="Rio de Janeiro">
    <block address="147.65.0.0" mask="16" />
    <block address="200.18.176.0" mask="20" />
    <block address="200.17.63.0" mask="24" />
    <block address="200.159.254.0" mask="24" />
    <block address="200.159.255.0" mask="24" />
    <block address="146.134.0.0" mask="16" />
    <block address="200.17.144.0" mask="20" />
  </range>
</blocks>
```

```
<block address="200.17.128.0" mask="20" />
<block address="200.18.224.0" mask="20" />
<block address="200.18.112.0" mask="20" />
<block address="200.128.0.0" mask="16" />
<block address="200.143.192.0" mask="18" />
<block address="200.159.240.0" mask="20" />
<block address="200.17.112.0" mask="20" />
<block address="200.17.48.0" mask="20" />
<block address="200.143.194.0" mask="24" />
</range>
```

```
<range name="Minas Gerais">
  <block address="200.18.144.255" mask="24" />
  <block address="200.18.144.0" mask="20" />
<block address="200.19.144.0" mask="20" />
  <block address="200.17.176.0" mask="20" />
  <block address="200.131.0.0" mask="16" />
  <block address="200.17.64.0" mask="20" />
  <block address="150.164.0.0" mask="16" />
  <block address="200.18.128.255" mask="24" />
  <block address="200.19.128.255" mask="24" />
  <block address="200.19.144.255" mask="24" />
  <block address="200.19.128.0" mask="20" />
  <block address="200.18.128.0" mask="20" />
  <block address="200.137.128.0" mask="18" />
</range>
```

```
<range name="Rio Grande do Sul">
  <block address="200.132.0.0" mask="25" />
</range>
```

```
<range name="Sao Paulo">
  <block address="200.185.4.0" mask="24" />
  <block address="200.235.1.192" mask="27" />
  <block address="200.136.34.0" mask="24" />
  <block address="200.17.61.0" mask="26" />
  <block address="200.136.30.0" mask="27" />
  <block address="143.108.254.152" mask="29" />
<block address="200.9.134.0" mask="24" />
</range>
```

```
<range name="Pernambuco">
  <block address="200.133.33.16" mask="28" />
  <block address="200.133.2.0" mask="24" />
  <block address="200.133.31.0" mask="24" />
  <block address="200.17.132.96" mask="27" />
  <block address="200.133.30.0" mask="24" />
  <block address="200.17.138.0" mask="24" />
  <block address="200.133.13.0" mask="24" />
  <block address="200.133.1.0" mask="24" />
</range>
```

```
<block address="200.139.0.0" mask="18" />
<block address="200.133.26.0" mask="24" />
<block address="200.133.23.0" mask="24" />
<block address="200.133.17.0" mask="24" />
<block address="200.137.166.0" mask="24" />
<block address="150.161.2.0" mask="24" />
<block address="200.139.0.96" mask="27" />
<block address="200.139.0.64" mask="27" />
<block address="200.139.10.0" mask="24" />
<block address="200.133.29.0" mask="24" />
<block address="200.17.133.0" mask="24" />
<block address="200.133.22.0" mask="24" />
<block address="200.133.33.80" mask="28" />
<block address="200.133.27.0" mask="24" />
<block address="200.139.4.0" mask="24" />
<block address="200.133.33.96" mask="28" />
<block address="200.133.34.0" mask="24" />
<block address="200.139.0.128" mask="27" />
<block address="200.139.8.0" mask="24" />
<block address="200.139.1.0" mask="24" />
<block address="192.231.114.0" mask="24" />
<block address="200.133.0.255" mask="24" />
<block address="150.161.0.0" mask="16" />
<block address="200.17.132.160" mask="27" />
<block address="200.17.132.192" mask="27" />
<block address="200.17.134.0" mask="24" />
<block address="200.17.137.0" mask="24" />
<block address="200.139.0.32" mask="27" />
<block address="200.17.132.128" mask="27" />
<block address="200.17.140.0" mask="24" />
<block address="200.133.33.48" mask="28" />
<block address="200.133.14.0" mask="24" />
<block address="200.238.126.0" mask="24" />
<block address="200.17.134.64" mask="27" />
<block address="150.161.42.0" mask="24" />
<block address="200.17.132.32" mask="27" />
<block address="200.238.127.0" mask="24" />
<block address="200.133.0.0" mask="16" />
<block address="200.139.12.0" mask="24" />
<block address="200.139.11.0" mask="24" />
<block address="200.17.139.0" mask="24" />
<block address="200.17.132.64" mask="27" />
</range>

<range name="Santa Catarina">
  <block address="200.135.15.0" mask="24" />
  <block address="150.135.0.0" mask="24" />
  <block address="150.162.0.0" mask="16" />
</range>
```

```
<range name="Paraiba">
  <block address="150.165.0.0" mask="16" />
  <block address="200.129.64.0" mask="18" />
</range>
```

```
<range name="Distrito Federal">
  <block address="200.130.34.0" mask="24" />
  <block address="200.214.130.192" mask="26" />
  <block address="200.252.232.192" mask="26" />
  <block address="200.129.240.0" mask="20" />
  <block address="200.130.32.0" mask="24" />
  <block address="200.19.119.0" mask="24" />
  <block address="200.18.192.0" mask="20" />
  <block address="200.252.232.128" mask="26" />
  <block address="200.19.112.0" mask="20" />
  <block address="200.18.208.0" mask="20" />
  <block address="200.18.16.0" mask="20" />
  <block address="200.17.0.0" mask="20" />
  <block address="200.18.80.0" mask="20" />
  <block address="200.130.33.0" mask="24" />
  <block address="200.129.128.0" mask="18" />
  <block address="200.130.0.0" mask="16" />
  <block address="200.19.16.0" mask="20" />
  <block address="200.19.32.0" mask="20" />
  <block address="200.18.192.0" mask="20" />
  <block address="200.19.116.0" mask="20" />
</range>
```

```
<range name="Rondonia">
  <block address="200.129.140.0" mask="24" />
  <block address="200.129.141.0" mask="24" />
  <block address="200.129.142.0" mask="24" />
  <block address="200.129.139.0" mask="24" />
</range>
```

```
</range>
```

```
<range name="Amapa">
  <block address="200.129.167.0" mask="24" />
</range>
```

```
<range name="Para">
  <block address="200.129.150.0" mask="24" />
  <block address="200.129.146.0" mask="24" />
  <block address="200.17.51.0" mask="24" />
  <block address="200.129.149.0" mask="24" />
  <block address="200.129.137.0" mask="24" />
  <block address="200.129.145.0" mask="24" />
  <block address="200.129.144.0" mask="24" />
  <block address="200.129.132.0" mask="24" />
  <block address="200.129.148.0" mask="24" />
  <block address="200.129.147.0" mask="24" />
</range>
```

```
<block address="200.129.128.0" mask="24" />
<block address="200.129.136.0" mask="24" />
<block address="200.129.138.0" mask="24" />
<block address="200.17.50.0" mask="24" />
</range>
```

```
<range name="Mato Grosso">
  <block address="200.129.240.0" mask="24" />
<block address="200.129.244.0" mask="24" />
<block address="200.17.60.0" mask="24" />
<block address="200.129.246.0" mask="24" />
<block address="200.129.242.0" mask="24" />
<block address="200.129.241.0" mask="24" />
</range>
```

```
<range name="Tocantins">
  <block address="200.129.176.0" mask="24" />
  <block address="200.129.177.0" mask="24" />
  <block address="200.129.178.0" mask="24" />
  <block address="200.129.179.0" mask="24" />
</range>
```

```
<range name="Amazonas">
  <block address="200.17.53.0" mask="24" />
  <block address="200.129.156.0" mask="24" />
  <block address="200.129.152.0" mask="24" />
  <block address="200.129.161.0" mask="24" />
  <block address="200.129.169.0" mask="24" />
  <block address="200.129.164.0" mask="24" />
  <block address="200.129.165.0" mask="24" />
  <block address="200.129.166.0" mask="24" />
  <block address="200.129.163.0" mask="24" />
  <block address="200.17.49.0" mask="24" />
  <block address="200.129.158.0" mask="24" />
  <block address="200.129.153.0" mask="24" />
  <block address="200.129.170.0" mask="24" />
  <block address="200.129.157.0" mask="24" />
  <block address="200.129.168.0" mask="24" />
</range>
```

```
<range name="Mato Grosso do Sul">
<block address="200.129.192.0" mask="20" />
  <block address="200.17.52.0" mask="24" />
</range>
```

```
<range name="Goias">
  <block address="200.17.57.0" mask="24" />
  <block address="200.17.55.0" mask="24" />
  <block address="200.18.160.0" mask="20" />
  <block address="200.137.192.0" mask="18" />
```

```
    <block address="200.17.58.0" mask="24" />
    <block address="200.17.56.0" mask="24" />
</range>
```

```
<range name="Piaui">
  <block address="200.137.160.0" mask="24" />
  <block address="200.137.172.0" mask="24" />
  <block address="200.137.179.0" mask="24" />
  <block address="200.137.164.0" mask="24" />
  <block address="200.137.178.0" mask="24" />
  <block address="200.137.176.0" mask="24" />
  <block address="200.137.177.0" mask="24" />
  <block address="200.137.174.0" mask="24" />
  <block address="200.137.165.0" mask="24" />
  <block address="200.137.175.0" mask="24" />
  <block address="200.137.171.0" mask="24" />
  <block address="200.137.167.0" mask="24" />
  <block address="200.137.162.0" mask="24" />
  <block address="200.137.169.0" mask="24" />
  <block address="200.137.163.0" mask="24" />
  <block address="200.137.170.0" mask="24" />
  <block address="200.137.173.0" mask="24" />
</range>
```

```
<range name="Maranhao">
  <block address="200.137.128.0" mask="24" />
  <block address="200.137.129.0" mask="25" />
  <block address="200.137.143.0" mask="24" />
  <block address="200.137.131.0" mask="24" />
  <block address="200.137.141.0" mask="24" />
  <block address="200.137.140.0" mask="24" />
  <block address="200.137.132.0" mask="24" />
  <block address="200.137.142.0" mask="24" />
  <block address="200.137.135.0" mask="24" />
  <block address="200.137.130.0" mask="24" />
  <block address="200.137.129.128" mask="25" />
  <block address="200.137.134.0" mask="24" />
  <block address="200.137.133.0" mask="24" />
  <block address="200.137.139.0" mask="24" />
</range>
```

```
<range name="Ceara">
  <block address="200.129.0.0" mask="18" />
  <block address="200.19.176.0" mask="20" />
  <block address="200.17.32.0" mask="20" />
  <block address="200.9.123.0" mask="24" />
  <block address="200.9.104.0" mask="24" />
</range>
```

```
<range name="Rio Grande do Norte">
```

```
<block address="200.137.0.0" mask="18" />
<block address="200.129.17.0" mask="26" />
<block address="200.17.143.0" mask="24" />
<block address="200.19.160.0" mask="20" />
</range>

<range name="Bahia">
  <block address="200.128.16.0" mask="24" />
  <block address="200.128.46.0" mask="24" />
  <block address="200.18.224.0" mask="24" />
  <block address="200.18.226.0" mask="24" />
  <block address="200.128.19.0" mask="24" />
  <block address="200.128.51.0" mask="24" />
  <block address="200.128.56.0" mask="24" />
  <block address="200.128.21.0" mask="24" />
  <block address="200.128.52.0" mask="24" />
  <block address="200.128.53.0" mask="24" />
  <block address="200.128.47.0" mask="24" />
  <block address="200.128.50.0" mask="24" />
  <block address="200.128.43.0" mask="24" />
  <block address="200.128.44.04" mask="24" />
  <block address="200.18.234.0" mask="24" />
  <block address="200.128.48.0" mask="24" />
  <block address="200.128.49.0" mask="24" />
  <block address="200.18.231.0" mask="24" />
  <block address="200.18.230.0" mask="24" />
  <block address="200.128.40.0" mask="24" />
  <block address="200.128.57.32" mask="27" />
  <block address="200.128.22.0" mask="24" />
  <block address="200.128.23.0" mask="24" />
  <block address="200.128.34.0" mask="24" />
  <block address="200.128.2.0" mask="24" />
  <block address="200.128.20.0" mask="24" />
  <block address="200.128.24.0" mask="24" />
  <block address="200.128.28.0" mask="24" />
  <block address="200.128.31.0" mask="24" />
  <block address="200.128.30.0" mask="24" />
  <block address="200.128.31.192" mask="24" />
  <block address="200.128.26.0" mask="24" />
  <block address="200.128.27.0" mask="24" />
  <block address="200.128.3.0" mask="24" />
  <block address="200.17.148.0" mask="24" />
  <block address="200.128.5.0" mask="24" />
  <block address="200.128.39.0" mask="24" />
  <block address="200.128.1.0" mask="24" />
  <block address="200.128.38.0" mask="24" />
  <block address="200.128.6.0" mask="24" />
  <block address="200.128.36.0" mask="24" />
  <block address="200.128.17.0" mask="24" />
  <block address="200.128.35.0" mask="24" />

```

```
<block address="200.128.8.0" mask="24" />
<block address="200.128.62.0" mask="24" />
<block address="200.128.10.0" mask="24" />
<block address="200.128.37.0" mask="24" />
<block address="200.17.146.0" mask="24" />
<block address="200.128.29.0" mask="24" />
<block address="200.128.42.0" mask="24" />
<block address="200.128.55.0" mask="24" />
<block address="200.128.4.0" mask="24" />
<block address="200.128.80.0" mask="25" />
<block address="200.128.60.64" mask="26" />
<block address="200.128.25.0" mask="24" />
<block address="200.18.228.0" mask="24" />
<block address="200.17.125.0" mask="24" />
<block address="200.128.61.0" mask="24" />
<block address="200.128.41.0" mask="24" />
<block address="200.128.18.0" mask="24" />
<block address="200.128.59.0" mask="24" />
<block address="200.18.227.0" mask="24" />
<block address="200.18.225.0" mask="24" />
<block address="200.128.54.0" mask="24" />
<block address="200.128.17.64" mask="26" />
<block address="200.128.63.0" mask="24" />
<block address="200.18.112.0" mask="24" />
<block address="200.18.113.0" mask="24" />
<block address="200.128.60.128" mask="26" />
<block address="200.128.80.128" mask="25" />
<block address="200.18.229.0" mask="24" />
<block address="200.128.9.0" mask="24" />
<block address="200.128.58.0" mask="24" />
<block address="200.128.57.128" mask="25" />
<block address="192.188.11.0" mask="24" />
<block address="200.18.117.0" mask="24" />
<block address="200.18.118.0" mask="24" />
<block address="200.18.115.0" mask="24" />
<block address="200.17.147.0" mask="24" />
<block address="200.17.149.0" mask="24" />
<block address="200.128.57.64" mask="27" />
<block address="200.17.144.0" mask="24" />
<block address="200.128.33.0" mask="24" />
</range>

<range name="Espirito Santo">
  <block address="200.137.64.0" mask="18" />
</range>

<range name="Alagoas">
<block address="200.133.127.0" mask="24" />
  <block address="200.17.112.0" mask="24" />
  <block address="200.17.113.0" mask="24" />
```

```
<block address="200.17.114.0" mask="24" />
<block address="200.17.116.0" mask="24" />
<block address="200.17.117.0" mask="24" />
<block address="200.133.126.0" mask="24" />
<block address="200.9.70.0" mask="24" />
<block address="200.17.115.0" mask="24" />
</range>
<range name="Roraima">
  <block address="200.129.143.248" mask="29" />
  <block address="200.129.143.0" mask="29" />
  <block address="200.129.143.8" mask="29" />
  <block address="200.129.159.0" mask="26" />
</range>

<range name="Sergipe">
  <block address="200.133.41.0" mask="24" />
  <block address="200.133.42.0" mask="24" />
  <block address="200.17.118.0" mask="24" />
  <block address="200.17.120.0" mask="24" />
  <block address="200.133.46.0" mask="24" />
  <block address="200.133.45.0" mask="24" />
  <block address="200.17.141.0" mask="24" />
  <block address="200.133.43.0" mask="24" />
  <block address="200.133.44.0" mask="24" />
</range>
</blocks>

<servers>
  <pool name="Fio Cruz" status="online">
    <server ip="157.86.124.177">
      <service name="dvod" port="9998" />
      <service name="dlive" port="6500" />
    </server>
  </pool>

  <pool name="Rio de Janeiro" status="online">
    <server ip="200.159.254.151">
      <service name="dvod" port="9998" />
      <service name="dlive" port="6500" />
    </server>
  </pool>

  <pool name="Rio Grande do Sul" status="online">
    <server ip="200.132.0.108">
      <service name="dvod" port="9998" />
      <service name="dlive" port="6500" />
    </server>
  </pool>

  <pool name="Sao Paulo" status="online">
```

```
        <server ip="200.17.61.33">
            <service name="dvod" port="9998" />
            <service name="dlive" port="6500" />
        </server>
    </pool>

    <pool name="Pernambuco" status="online">
        <server ip="200.133.0.251">
            <service name="dvod" port="9998" />
            <service name="dlive" port="6500" />
        </server>
    </pool>

    <pool name="Santa Catarina" status="online">
        <server ip="200.135.0.12">
            <service name="dvod" port="9998" />
            <service name="dlive" port="6500" />
        </server>
    </pool>

    <pool name="Paraiba" status="online">
        <server ip="150.165.132.1">
            <service name="dvod" port="9998" />
            <service name="dlive" port="6500" />
        </server>
    </pool>

    <pool name="Distrito Federal" status="online">
        <server ip="200.19.119.112">
            <service name="dvod" port="9998" />
            <service name="dlive" port="6500" />
        </server>
    </pool>

    <pool name="Ceara" status="online">
        <server ip="200.129.0.80">
            <service name="dvod" port="9998" />
            <service name="dlive" port="6500" />
        </server>
    </pool>

    <pool name="Rio Grande do Norte" status="online">
        <server ip="200.19.162.17">
            <service name="dvod" port="9998" />
            <service name="dlive" port="6500" />
        </server>
    </pool>
</servers>
</netdescr>
```

Apêndice III

Documento de Validação (Arquivo XML Schema)

Este Apêndice descreve o documento de validação completo que está em funcionamento no sistema de distribuição de vídeo digital da rede RNP.

```
<!-- edited with XMLSpy v2005 rel. 3 U (http://www.altova.com) by Fernando
(Pessoal) -->
<!--W3C Schema generated by XMLSPY v2004 rel. 4 U (http://www.xmlspy.com)-->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified">

  <xs:element name="netnodes">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="node" maxOccurs="unbounded">
          <xs:complexType>
            <xs:attribute name="name" use="required">
              <xs:simpleType>
                <xs:restriction base="xs:string">
                  <xs:whiteSpace value="preserve"/>
                  <xs:pattern value="([A-Za-z ]+"/>
                </xs:restriction>
              </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="local" use="required">
              <xs:simpleType>
                <xs:restriction base="xs:string">
                  <xs:whiteSpace value="preserve"/>
                  <xs:pattern value="([a-zA-Z ]+"/>
                </xs:restriction>
              </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="latitude" use="required">
              <xs:simpleType>
                <xs:restriction base="xs:decimal">
                </xs:restriction>
              </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="longitude" use="required">
              <xs:simpleType>
                <xs:restriction base="xs:string">
                </xs:restriction>
              </xs:simpleType>
            </xs:attribute>
```

```
<xs:attribute name="default" use="optional">
<xs:simpleType>
<xs:restriction base="xs:string">
<xs:pattern value="yes|no"/>
</xs:restriction>
</xs:simpleType>
</xs:attribute>
<xs:attribute name="hasServer" use="required">
<xs:simpleType>
<xs:restriction base="xs:string">
<xs:pattern value="yes|no"/>
</xs:restriction>
</xs:simpleType>
</xs:attribute>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="connections">
<xs:complexType>
<xs:sequence>
<xs:element name="source" maxOccurs="unbounded">
<xs:complexType>
<xs:sequence>
<xs:element name="target" maxOccurs="unbounded">
<xs:complexType>
<xs:attribute name="name" use="required">
<xs:simpleType>
<xs:restriction base="xs:string">
<xs:whiteSpace value="preserve"/>
<xs:pattern value="([a-zA-Z ]+"/>
</xs:restriction>
</xs:simpleType>
</xs:attribute>
<xs:attribute name="rate" type="xs:integer" use="required"/>
<xs:attribute name="unit" use="required">
<xs:simpleType>
<xs:restriction base="xs:string">
<xs:enumeration value="bps"/>
<xs:enumeration value="Kbps"/>
<xs:enumeration value="Mbps"/>
<xs:enumeration value="Gbps"/>
</xs:restriction>
</xs:simpleType>
</xs:attribute>
</xs:complexType>
</xs:element>
</xs:sequence>
<xs:attribute name="name" type="xs:string" use="required"/>
```

```
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="blocks">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="range" maxOccurs="unbounded">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="block" maxOccurs="unbounded">
              <xs:complexType>
                <xs:attribute name="address" use="required">
                  <xs:simpleType>
                    <xs:restriction base="xs:string">
                      <xs:pattern value="((([0-2][0-5][0-5][0-2][0-4][0-9][1][0-9][0-9][0-9][0-9])\.[0-2][0-5][0-5][0-2][0-4][0-9][1][0-9][0-9][0-9][0-9])\.[0-2][0-5][0-5][0-2][0-4][0-9][1][0-9][0-9][0-9][0-9])\.[0-2][0-5][0-5][0-2][0-4][0-9][1][0-9][0-9][0-9][0-9])\.[0-2][0-5][0-5][0-2][0-4][0-9][1][0-9][0-9][0-9][0-9])"/>
                    </xs:restriction>
                  </xs:simpleType>
                </xs:attribute>
                <xs:attribute name="mask" use="required">
                  <xs:simpleType>
                    <xs:restriction base="xs:integer">
                      <xs:maxInclusive value="32"/>
                      <xs:minInclusive value="0"/>
                    </xs:restriction>
                  </xs:simpleType>
                </xs:attribute>
              </xs:complexType>
            </xs:element>
          </xs:sequence>
          <xs:attribute name="name" use="required">
            <xs:simpleType>
              <xs:restriction base="xs:string">
                <xs:whiteSpace value="preserve"/>
                <xs:pattern value="([a-zA-Z ]+"/>
              </xs:restriction>
            </xs:simpleType>
          </xs:attribute>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="servers">
  <xs:complexType>
    <xs:sequence>
```



```
<xs:pattern value="(online|offline)"/>
</xs:restriction>
</xs:simpleType>
</xs:attribute>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="netdescr">
<xs:complexType>
<xs:sequence>
<xs:element ref="netnodes"/>
<xs:element ref="connections"/>
<xs:element ref="blocks"/>
<xs:element ref="servers"/>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:schema>
```

Livros Grátis

(<http://www.livrosgratis.com.br>)

Milhares de Livros para Download:

[Baixar livros de Administração](#)

[Baixar livros de Agronomia](#)

[Baixar livros de Arquitetura](#)

[Baixar livros de Artes](#)

[Baixar livros de Astronomia](#)

[Baixar livros de Biologia Geral](#)

[Baixar livros de Ciência da Computação](#)

[Baixar livros de Ciência da Informação](#)

[Baixar livros de Ciência Política](#)

[Baixar livros de Ciências da Saúde](#)

[Baixar livros de Comunicação](#)

[Baixar livros do Conselho Nacional de Educação - CNE](#)

[Baixar livros de Defesa civil](#)

[Baixar livros de Direito](#)

[Baixar livros de Direitos humanos](#)

[Baixar livros de Economia](#)

[Baixar livros de Economia Doméstica](#)

[Baixar livros de Educação](#)

[Baixar livros de Educação - Trânsito](#)

[Baixar livros de Educação Física](#)

[Baixar livros de Engenharia Aeroespacial](#)

[Baixar livros de Farmácia](#)

[Baixar livros de Filosofia](#)

[Baixar livros de Física](#)

[Baixar livros de Geociências](#)

[Baixar livros de Geografia](#)

[Baixar livros de História](#)

[Baixar livros de Línguas](#)

[Baixar livros de Literatura](#)
[Baixar livros de Literatura de Cordel](#)
[Baixar livros de Literatura Infantil](#)
[Baixar livros de Matemática](#)
[Baixar livros de Medicina](#)
[Baixar livros de Medicina Veterinária](#)
[Baixar livros de Meio Ambiente](#)
[Baixar livros de Meteorologia](#)
[Baixar Monografias e TCC](#)
[Baixar livros Multidisciplinar](#)
[Baixar livros de Música](#)
[Baixar livros de Psicologia](#)
[Baixar livros de Química](#)
[Baixar livros de Saúde Coletiva](#)
[Baixar livros de Serviço Social](#)
[Baixar livros de Sociologia](#)
[Baixar livros de Teologia](#)
[Baixar livros de Trabalho](#)
[Baixar livros de Turismo](#)