WALTER AUGUSTO VARELLA

**Laboratório Remoto em Ambiente Aberto com
Protocolo IPv6**

Dissertação apresentada à Escola
Politécnica da Universidade de São
Paulo para obtenção do título de
Mestre em Engenharia Elétrica.

São Paulo

2003

**CONSULTA
FD-3474**

OK

WALTER AUGUSTO VARELLA

**Laboratório Remoto em Ambiente Aberto com
Protocolo IPv6**

**Dissertação apresentada à Escola Politécnica
da Universidade de São Paulo para obtenção
do Título de Mestre em Engenharia Elétrica.**

**Área de Concentração:
Microeletrônica**

**Orientador:
Prof. Dr. Francisco Javier Ramirez Fernandez**

**São Paulo
2003**

Varella, Walter Augusto.

Laboratório Remoto em Ambiente Aberto com Protocolo IPv6, São Paulo, 2003.

120 p.

Dissertação (Mestrado) – Escola Politécnica da Universidade de São Paulo. Departamento de Sistemas Eletrônicos.

1. Instrumentação Virtual 2. Laboratório Virtual 3. Monitoração Remota 4. Protocolo IP

1. Universidade de São Paulo. Escola Politécnica. Departamento de Sistemas Eletrônicos.

BC

Universidade de São Paulo
Biblioteca da Escola Politécnica

FD- 3474

“Eu irei adiante de ti, endireitarei os caminhos
tortuosos, quebrarei as portas de bronze e
despedaçarei as trancas de ferro;
dar-te-ei os tesouros escondidos e as riquezas
encobertas, para que saibas que eu sou o
SENHOR, o Deus de Israel, que te chama pelo
teu nome”. Is 45,2-3

AGRADECIMENTOS

Agradeço a Deus pela vida e as condições que me proporcionaram ser feliz e desfrutar de todos os momentos, sonhar e adquirir a sabedoria que vem do alto.

Ao meu orientador e amigo Prof. Dr. Francisco Javier, em quem observei profundo senso de respeito ao próximo e suas limitações. A sua liderança e a maneira de conduzir um grupo de pesquisa me ensinaram muito no relacionamento com as pessoas.

Aos meus pais que me ensinaram a andar no caminho certo, e que hoje compartilham mais uma vitória.

Agradeço a minha esposa, Maria do Carmo, pelo carinho, amor, confiança e dedicação, que reafirmam a cada dia ser a mulher da minha vida.

Aos meus filhos, Thiago e Daniel, que colaboram a cada momento com a minha vida e felicidade.

Aos amigos, Pr. João Campos e sua esposa Annika, pelo carinho e as palavras de incentivo nos momentos difíceis.

Aos irmãos e amigos da Igreja Batista Filadélfia e do 100% Vida São Vicente, pelas orações em meu favor.

Ao amigo Carlos Rebolledo pela colaboração em muitos momentos.

Aos colegas de pesquisa do Grupo SIM, que com suas experiências sempre estiveram presentes.

Aos adolescentes e amigos, Thiago Lima, André, Maurinho, Mayra, Flaviana e Luis Henrique pelo amor demonstrado nas horas difíceis.

Aos meus alunos, do Centro Federal de Educação Tecnológica de São Paulo, pela colaboração e apoio nos sonhos de projetos do Laboratório Virtual.

Aos diretores do CEFET – SP e da Unidade de Cubatão pelo incentivo dado à pesquisa e pelo espírito empreendedor desenvolvido na administração de uma escola pública.

Aos amigos Pr. Rune Söderberg e esposa Ulla-Britt pelo exemplo de projeto de vida que deu certo em seus cinquenta anos de união.

A National Instruments e sua equipe de suporte no Brasil, pela qualidade de atendimento e colaboração no desenvolvimento da pesquisa.

RESUMO

Nesta dissertação são apresentados os resultados obtidos na elaboração de um sistema de aquisição e controle de experimentos, utilizando uma rede local de computadores. É proposta uma infra-estrutura baseada em instrumentação virtual em uma rede utilizando sistema operacional aberto e o protocolo de comunicação IPv6, parte integrante da camada de rede do protocolo TCP/IP. São selecionadas as tecnologias e ferramentas para o desenvolvimento do Laboratório remoto que se adaptam à integração entre o hardware e software para possibilitar o controle de experimentos e equipamentos de forma remota. A maior abrangência deste projeto está no desenvolvimento do Laboratório remoto com ferramentas de código aberto, como o sistema operacional Linux, o servidor Apache - como servidor WEB e o protocolo IPv6, para a comunicação da camada de rede.

A proposta global deste trabalho é contribuir com a preparação da infra-estrutura de um ambiente de sistema operacional aberto, utilizado para a aquisição de dados em redes remotas, e com isto promover a cooperação e o compartilhamento de instrumentos, não se limitando apenas à área acadêmica e de pesquisa, mas estendendo essas facilidades para outras áreas do conhecimento.

Como resultado deste trabalho é disponibilizado um Laboratório Virtual que possibilita o controle de um ensaio experimental, através de uma rede local.

ABSTRACT

In this dissertation are showed the results got in the making of acquisition and controlling experiments system, using a LAN. It's offered a infra-structure based in virtual instrumentation for a network applying open operational system and the communication protocol Ipv6, part which belongs to network layer of TCP/IP protocol.

The technologies and tools, which are chosen for the development of remote Laboratory, adapt themselves to integration between hardware and software to making possible the experiments and equipments control remoting.

The main purpose of this project consist in the development of remote Laboratory with open-code tools, as Linux operational system, Apache server – as WEB server and Ipv6, for network layer communication.

The global proposal of this work is contribute to setting up infra-structure of an open operational system, utilized to data acquisition in remote network, promote the work jointly and sharing of instruments, not restricting only to academic or knowledge areas.

Like resulting of this work is available a Virtual Laboratory, which allow the control of experimental assay, through a LAN.

SUMÁRIO

LISTA DE FIGURAS	
LISTA DE TABELAS	
LISTA DE ABREVIATURAS E SIGLAS	
RESUMO	
ABSTRACT	

1	INTRODUÇÃO.....	1
2	LABORATÓRIO VIRTUAL	5
2.1	INSTRUMENTAÇÃO VIRTUAL	5
2.1.1	<i>Paradigma da programação utilizada no Laboratório Virtual.....</i>	<i>6</i>
2.1.2	<i>Hardware para instrumentação virtual.....</i>	<i>7</i>
2.1.3	<i>Redes de Computadores.....</i>	<i>8</i>
2.1.4	<i>Internet e os serviços disponíveis.....</i>	<i>9</i>
2.1.4.1	O protocolo FTP	10
2.1.4.2	O protocolo TELNET	11
2.1.4.3	O protocolo SMTP.....	11
2.1.4.4	O protocolo HTTP.....	11
2.1.5	<i>Sistemas Cliente/Servidor</i>	<i>13</i>
2.1.6	<i>Considerações finais.....</i>	<i>14</i>
3	A FERRAMENTA DE SOFTWARE - LABVIEW®	16
3.1	CARACTERÍSTICAS BÁSICAS DO LABVIEW®	16
3.2	ESTRUTURAS DE PROGRAMAÇÃO EM LABVIEW®	18
3.3	INTERFACES DE AQUISIÇÃO DE DADOS.....	21
3.3.1	<i>Comunicação Serial</i>	<i>22</i>
3.3.2	<i>Arquitetura de Software de Instrumentos Virtuais -VISA.....</i>	<i>25</i>
3.3.3	<i>Barramento de Interface de uso geral - GPIB.....</i>	<i>28</i>
3.3.4	<i>Aquisição de dados – DAQ.....</i>	<i>34</i>
3.4	PROTOCOLOS DE COMUNICAÇÃO.....	37
3.4.1	<i>Protocolo DataSocket®</i>	<i>38</i>
3.4.2	<i>Características e limitações do Laboratório Virtual com o DataSocket®</i>	<i>40</i>
3.4.3	<i>Velocidade de transferência.....</i>	<i>40</i>
3.4.4	<i>Tamanho dos pacotes de dados.....</i>	<i>41</i>
3.4.5	<i>Formato de dados suportados pelo DataSocket®</i>	<i>41</i>
3.4.6	<i>Criptografia e segurança de dados</i>	<i>41</i>
3.5	O LABVIEW® E O PROTOCOLO TCP/IP	42
3.6	CONSIDERAÇÕES FINAIS	45
4	HISTÓRICO DO PROTOCOLO IP.....	46
4.1	CONCEITOS BÁSICOS SOBRE PROTOCOLO IP.....	46
4.2	O PROTOCOLO IPV6 – CONCEITOS.....	48
4.3	CABEÇALHOS DOS PROTOCOLOS IPV4 E IPV6.....	50
4.3.1	<i>Cabeçalhos de extensão do protocolo IPv6.....</i>	<i>54</i>
4.3.2	<i>Cabeçalho de opção hop a hop.....</i>	<i>57</i>

4.3.3	<i>Cabeçalho de opções de destino</i>	58
4.3.4	<i>Cabeçalho de roteamento</i>	59
4.3.5	<i>Cabeçalho de fragmentação</i>	60
4.3.6	<i>Cabeçalho de Autenticação</i>	61
4.3.7	<i>Cabeçalho de carga de segurança de encapsulamento</i>	62
4.4	ARQUITETURA DE ENDEREÇAMENTO DO PROTOCOLO IPV6	64
4.5	NOTAÇÃO DE ENDEREÇOS IPV6	65
4.6	HIERARQUIA DOS ENDEREÇOS IPV6	67
4.6.1	<i>Endereços Unicast</i>	69
4.6.2	<i>Endereço Anycast</i>	72
4.6.3	<i>Endereço Multicast</i>	73
4.7	TRANSMISSÃO DE PACOTES IPV6 EM REDES ETHERNET	76
4.8	CENÁRIOS DE TRANSIÇÃO DO PROTOCOLO IPV4 PARA IPV6	77
4.8.1	<i>Cenário de Transição – utilização da pilha dupla IPv4/IPv6</i>	77
4.8.2	<i>Cenário de transição - Tunelamento de IPv6 sobre IPv4</i>	78
4.8.3	<i>Tunelamento sobre IPv4 configurado ou estático</i>	80
4.8.4	<i>Tunelamento sobre IPv4 automático</i>	81
4.8.5	<i>Tunelamento IPv6</i>	82
4.9	CONSIDERAÇÕES FINAIS	83
5	MATERIAIS E MÉTODOS	84
5.1	OS MATERIAIS.....	84
5.1.1	<i>Os equipamentos - Hardware</i>	84
5.2	OS PROGRAMAS UTILIZADOS – SOFTWARE	85
5.2.1	<i>O Sistema Operacional – Linux</i>	86
5.2.2	<i>O servidor WEB – Apache</i>	88
5.2.3	<i>O protocolo IPv6</i>	89
5.3	ETAPAS PRÉ-ESTABELECIDAS PARA O PROJETO	89
5.4	METODOLOGIA ADOTADA.....	90
5.5	CONSIDERAÇÕES FINAIS.....	93
6	RESULTADOS EXPERIMENTAIS	94
6.1	ENSAIO EXPERIMENTAL.....	94
6.2	DIODO	96
6.3	DISPOSITIVOS FOTODETETORES	98
6.3.1	<i>Fotoresistência</i>	98
6.3.2	<i>Fotodiodo</i>	99
6.3.3	<i>Fototransistor</i>	100
6.4	PRIMEIRA ETAPA – SELEÇÃO DO ENSAIO EXPERIMENTAL	101
6.5	SEGUNDA ETAPA – IMPLEMENTAÇÃO DO INSTRUMENTO VIRTUAL	102
6.5.1	<i>Implementação do VI do Servidor</i>	103
6.5.2	<i>Implementação do VI Cliente</i>	105
6.6	CONCLUSÕES PARCIAIS	108
7	RESULTADOS EXPERIMENTAIS – AMBIENTE DE REDE	109
7.1	IMPLEMENTAÇÃO DO AMBIENTE DE REDE	109
7.1.1	<i>Configuração da interface de rede com o protocolo IPv6</i>	111
7.1.2	<i>Configuração do ambiente de rede</i>	112
7.1.3	<i>Testes de comunicação na rede IPv6</i>	114

7.2	IMPLEMENTAÇÃO DO CONTROLE ATRAVÉS DA REDE LOCAL	115
7.2.1	<i>Software de Servidor WEB – Apache</i>	116
7.2.1.1	Instalação do software Apache	116
7.2.1.2	Configuração do servidor Apache	117
7.3	IMPLEMENTAÇÃO DO LABORATÓRIO REMOTO EM IPV6	120
7.4	CONCLUSÕES PARCIAIS	121
8	CONCLUSÕES E PERSPECTIVAS FUTURAS.....	122
	REFERÊNCIAS BIBLIOGRÁFICAS.....	124

LISTA DE FIGURAS

Figura 1 – Exemplo de um sistema de instrumentação virtual.	6
Figura 2 – Aquisição, análise e apresentação.....	7
Figura 3 – Protocolos utilizados nas diversas camadas do TCP/IP.....	10
Figura 4 – Comportamento de requisição-resposta do protocolo HTTP.....	13
Figura 5 – Arquitetura de uma rede cliente/servidor e o controle dos instrumentos..	14
Figura 6 – Painel frontal de uma aplicação em LabVIEW®.....	17
Figura 7 – Diagrama de blocos de uma aplicação em LabVIEW®.....	18
Figura 8 – Bloco iterativo for.....	19
Figura 9 – Bloco iterativo “while”.....	19
Figura 10 – Estrutura do bloco “case” nas condições True e False.....	20
Figura 11 – Bloco de estrutura de sequência.....	20
Figura 12 – Exemplo de um diagrama utilizando as estruturas de repetição e de sequência.	20
Figura 13 – Comunicação serial entre microcomputador e um instrumento.....	22
Figura 14 – Transmissão serial do caractere “m”.....	22
Figura 15 – Tabela de caracteres ASCII.....	23
Figura 16 – Porta serial RS-232 de nove pinos.....	24
Figura 17 – Arquitetura VISA e os dispositivos de comunicação.....	25
Figura 18 – VISA Write: escrever dado em um dispositivo especificado.....	26
Figura 19 - VISA Read: lê dados de um dispositivo especificado.....	27
Figura 20 – Diagrama de blocos e painel frontal para um VI de teste de loopback serial.....	27
Figura 21 – Diagrama de blocos e painel frontal para um VI de um Simulador de escrita e leitura.....	28
Figura 22 – Placa PCI-GPIB com cabo da National Instruments.....	29
Figura 23 – Comunicação entre instrumentos e computador utilizando GPIB.....	29
Figura 24 – Configuração estrela e configuração linear de equipamentos com GPIB	30
Figura 25 – Especificações do conector GPIB.....	30
Figura 26 – Hierarquia típica de um <i>driver</i> de instrumento.....	32
Figura 27 – Painel frontal para o VI Histórico de tensão.....	34
Figura 28 – Diagrama de blocos para o VI Histórico de tensão.....	34
Figura 29 – Painel frontal de um instrumento voltímetro.....	36
Figura 30 – Diagrama de blocos do VI do voltímetro.....	36
Figura 31 – Painel frontal do VI utilizando controle de buffer e taxa de varredura ..	37
Figura 32 – Diagrama de blocos do VI utilizando controle de buffer e taxa de varredura.....	37
Figura 33 – Utilização do DataSocket® para compartilhamento de dados, através de aplicação em LabVIEW® e a publicação em uma página WEB.	39
Figura 34 – Diagrama de um VI com escrita e leitura, utilizando DataSocket®.....	40
Figura 35 – Comunicação TCP/IP utilizada no LabVIEW®.....	43
Figura 36 – Diagrama com detalhamento da VI TCP Listen.....	43
Figura 37 – Controle de múltiplas conexões com TCP/IP.....	44
Figura 38 – Possibilidades de comandos para a estrutura <i>case</i> do VI.....	45
Figura 39 - Cabeçalho do protocolo IPv4.....	50

Figura 40 – Formato do datagrama IPv6	52
Figura 41 - Cabeçalho do protocolo IPv6	53
Figura 42 – Exemplo de inserção de cabeçalho de extensão	55
Figura 43 – Sequência dos cabeçalhos de extensão no pacote IPv6	56
Figura 44 – TLV de um cabeçalho de extensão	57
Figura 45 – Cabeçalho de opções hop a hop	58
Figura 46 – Formato do cabeçalho de opção de destino	59
Figura 47 – Formato do cabeçalho de roteamento tipo 0	60
Figura 48 – Fragmentação de pacote por nó de origem do IPv6	61
Figura 49 – Formato do cabeçalho de autenticação (AH)	61
Figura 50 – Formato do cabeçalho de segurança de encapsulamento	63
Figura 51- Formato do Prefixo Provider – Based Unicast	66
Figura 52 - Formato do Aggregatable Global Unicast Address	67
Figura 53 – Técnica para atribuir endereços IPv6, de forma hierárquica	69
Figura 54 – Estrutura de endereço unicast recebido por nó e por um roteador	70
Figura 55 – Endereço IPv6 mapeado para IPv4	70
Figura 56– Formato do endereço do link local	71
Figura 57- Formato do endereço unicast do Site local	71
Figura 58 - Formato do endereço subnet-router anycast address	73
Figura 59- Formato do endereço multicast do IPv6	74
Figura 60 – Exemplo de endereço multicast	75
Figura 61 – Transmissão IPv6 através de redes Ethernet	76
Figura 62 – Implementação de pilha dupla IPv4/IPv6	78
Figura 63 – Encapsulamento do datagrama IPv6 no pacote IPv4	79
Figura 64 – Tunelamento de datagrama IPv6 sobre IPv4 configurado	80
Figura 65 – Tunelamento IPv4 automático	81
Figura 66 – Tunelamento IPv6 unidirecional	82
Figura 67 – Tunelamento IPv6 bidirecional	82
Figura 68 – Circuito eletrônico utilizado no ensaio do Laboratório Virtual	85
Figura 69 – Etapas a serem desenvolvidas para a realização deste trabalho	90
Figura 70– Estrutura do sistema de comunicação e aquisição de dados proposto para o projeto	92
Figura 71 – Metodologia de desenvolvimento adotada para o projeto	92
Figura 72 – Espectro eletromagnético com as faixas de comprimento de ondas	95
Figura 73 – Curva característica de diodos	97
Figura 74 – Curva características do fotodiodo	98
Figura 75 – Uma junção PN sendo iluminada e a geração de pares de elétrons e lacunas	99
Figura 76 – Uma junção PN não iluminada e o efeito da recombinação	99
Figura 77 – Símbolo e curva característica de um fotodiodo	100
Figura 78 – Símbolo e curvas características de um fototransistor	101
Figura 79 – Diagrama de blocos do ensaio experimental	102
Figura 80 – Circuito eletrônico utilizado no ensaio experimental	102
Figura 81 – Diagrama de blocos do VI implementado no servidor	104
Figura 82 – Painel frontal disponível no Servidor do Experimento	105
Figura 83 – Diagrama de blocos do Sub-VI DataReceive	105
Figura 84 – Diagrama de blocos do VI implementado no cliente	106
Figura 85 – Diagrama de blocos do Sub VI DATASend	107

Figura 86 - Painel frontal disponível no cliente	107
Figura 87 – Módulos carregados no kernel do Linux.....	110
Figura 88 – Comando ifconfig e a configuração da placa de rede	111
Figura 89 – Arquivo “protocols”, utilizado para configuração dos cabeçalhos de extensão	113
Figura 90 - Arquivo “hosts”, utilizado para configuração dos tipos de endereços ..	114
Figura 91 – Teste de comunicação entre estações com o comando ping6.....	115
Figura 92 – Configuração do Servidor WEB com o LabVIEW®	120
Figura 93 – Endereçamento do servidor WEB publicado com o LabVIEW®	121

LISTA DE TABELAS

Tabela 1 – Terminais do conector serial de um microcomputador padrão PC-IBM	24
Tabela 2- Espaço de endereçamento do protocolo IPv4.....	46
Tabela 3 - Características do campo Opções do protocolo IPv4.....	52
Tabela 4- Prefixos de endereços utilizados no IPv6.....	66
Tabela 5 – Endereços multicast reservados	75

LISTA DE ABREVIATURAS E SIGLAS

ADC	Analog-to-Digital Converter
ANSI	American National Standards Institute
API	Application Program Interface
ARPA	Advanced Research Projects Agency
ARPANET	Advanced Research Projects Agency Network
ASCII	American Standard Code for Information Interchange
ATN	Attention
AT-MIO	Advanced Technology Multiple Input Output
AH	Authentication Header
CIDR	Classless Inter-Domain Routing
CGI	Common Graphics Interface
CLNP	Connectionless Network Protocol
DAQ	Data Acquisition
DAV	Data Valid
DMA	Direct Memory Access
EOI	End or Identify
ESP	Encrypted Security Payload
FP	Format Prefix
GIF	Graphics Interchange Format
GSIM	Grupo Sensores Integráveis e Microsistemas
GPB	General Purpose Interface Bus
HTML	HyperText Markup Language
HTTP	Hypertext Transfer Protocol
ICMP	Internet Control Message Protocol
ICV	Integrity Check Value
ID	Identification
IEEE	Institute of Electrical and Electronics Engineers
IESG	Internet Engineering Steering Group
IETF	Internet Engineering Task Force
IFC	Interface Clear
IHL	Internet Header Length
IP	Internet Protocol
IPng	Internet Protocol next generation
IPv4	Internet Protocol version quatro
IPv6	Internet Protocol version seis
IPX	Internet Packet Exchange
IRQ	Interrupt Request
IV	Initialization Vector
JPEG	Joint Photographic Experts Group
LDR	Light dependent resistor
LLC	Logical Link Control
MEC	Ministério da Educação e Cultura
MTU	Maximum Transfer Unit
NDAC	Not data accepted
NLA	Next- Level Aggregation Identifier

NRFD	Not Ready for data
NSAP	Network Service Access Point
OSI	Open Systems Interconnection
PCI	Peripheral Component Interconnect
PXI	PCI eXtensions for Instrumentation
QoS	Quality of Service
REN	Remote Enable
RES	Reservado
RFC	Request for Comment
SEMTEC	Secretaria de Educação Média e Tecnológica
SIM	Sensores Integráveis e Microsistemas
SIPP	Simple Internet Protocol Plus
SLA	Site-Level Aggregation Identifier
SPI	Secure Parameter Index
SSI	Server Side Include
TCP	Transfer Control Protocol
TCP/IP	Transfer Control Protocol/Internet Protocol
TLA	Top-Level Aggregation Identifier
TLV	Type-Length-Values
URL	Uniform Resource Locator
USB	Universal Serial Bus
VXI	VMEbus eXtensions for instrumentation
WWW	World Wide Web

1 INTRODUÇÃO

As instituições de ensino, cujos cursos apresentam necessidades de experimentação laboratorial, têm enfrentado muitas dificuldades quanto à qualidade de ensino. Se por um lado, a utilização de novas tecnologias, aliada aos sistemas de comunicação, proporciona um avanço sem precedentes na área da educação e a difusão rápida do conhecimento, por outro exige um maior investimento na construção de laboratórios, com soluções específicas para um determinado curso ou até mesmo para uma disciplina. O MEC – Ministério da Educação e Cultura [1] e a SEMTEC - Secretaria de Educação Média e Tecnológica [2], nos documentos que norteiam a criação e a avaliação de cursos, estabelecem indicadores de qualidade para a relação alunos X posto de trabalho, o que leva as instituições de ensino, tanto a nível técnico quanto tecnológico, a buscar soluções que possam atender as exigências legais e estas soluções, em geral, elevam o custo do investimento em laboratórios, recaindo nos altos preços das mensalidades.

Algumas soluções adotadas para atender as exigências do MEC e da SEMTEC são de ordem administrativa, dividindo a turma para melhorar a relação de alunos e equipamentos e assim alcançar um melhor aproveitamento didático, mas ao custo de mais professores e dificuldades de alocação de horários. Uma outra solução é o desenvolvimento de parcerias cooperadas, para a construção e manutenção dos laboratórios. Ainda há de se considerar uma outra possibilidade, que é ignorar o problema e manter os laboratórios com vários alunos, poucos equipamentos e qualidade de aprendizado prático muito baixa.

Ao analisar os equipamentos de um laboratório, encontramos uma possibilidade de redução dos investimentos necessários com licenças de software, quer seja nos sistemas operacionais ou até mesmo nos programas de controle e aplicativos. A principal contribuição deste trabalho está ligada à redução dos investimentos com soluções proprietárias de software, utilizando onde possível, as soluções de software livre, sem os custos das licenças e manutenções anuais.

Ainda devemos considerar neste contexto, as necessidades do mercado e do mundo do trabalho [2], pois as instituições de ensino buscam, cada vez mais, aproximarem-se dele e manter uma sintonia com ele, o que tem refletido nas políticas

e práticas pedagógicas. Outro fator a ser considerado é a condição do país, em que a maior fonte geradora de renda está ligada às pequenas e médias empresas, com dificuldades de investimento e atualização tecnológica. Este cenário proporciona às instituições de ensino a pesquisa e o desenvolvimento de produtos e serviços utilizando ferramentas de software livre.

As soluções tecnológicas residem, então, em compartilhar os recursos dos laboratórios através da utilização de computadores, interligados em redes, com um conjunto de programas que pretendem ensinar atividades laboratoriais com técnicas de simulação, e o uso de recursos multimídia para apresentação dos conteúdos. Uma outra forma consiste em criar Laboratórios Virtuais [3], [4], diminuindo o custo com aquisição de equipamentos, e disponibilizando as atividades práticas com a utilização de placas de aquisição de dados e a implementação dos experimentos didáticos através de software.

A instrumentação virtual foi originalmente concebida como sendo uma ferramenta que possibilita o desenvolvimento de automatização nos processos laboratoriais e industriais. A instrumentação convencional, onerosa, não satisfazendo as necessidades de se ter controles de dados e sinais ao mesmo tempo [5], possibilitaram que a instrumentação virtual fosse inserida nos computadores pessoais com a utilização de placas para a aquisição de dados [6], oferecendo flexibilidade na monitoração e controle de processos na área industrial, na pesquisa e na eletrônica embarcada [7].

A evolução das redes de computadores, a utilização de ferramentas para a criação de laboratórios virtuais aplicados em aulas práticas, onde a experimentação e manipulação de instrumentos de medida e aquisição de dados são uma forma didática recente e aplicada em diversas universidades, como por exemplo, Universidade Nacional de Singapura –NUS- [8], Universidade de Stanford [9], Universidade de Widener [10], Instituto de Tecnologia Stevens [11], Universidade de Padova [12], Universidade do Estado de Portland [13], Universidade Politécnica de Valência [14] proporcionando o intercâmbio de informações entre grupos de pesquisa, situados em diferentes áreas geográficas.

Este trabalho, como proposta de solução tecnológica, propõe a utilização de software livre para a base operacional, no caso, o Sistema Operacional Linux [15],

que além do baixo custo de propriedade, oferece segurança, performance, liberdade de criação de aplicações devido ao seu código aberto, além de uma comunidade de desenvolvedores espalhados por todo o mundo, criando e adaptando soluções através de modelos de trabalhos colaborativos ou cooperativos.

Para a interligação das redes de computadores, o protocolo TCP/IP atualmente na versão quatro é o mais utilizado, permitindo a interoperabilidade entre diversas plataformas de sistemas operacionais, compartilhamento de informações e recursos de hardware entre os equipamentos participantes da rede. Com a globalização e a demanda crescente por serviços utilizando a Internet, o problema da escassez de endereços se torna cada vez mais evidente e para solucioná-lo, dentre outras propostas iniciais, surgem pesquisas para a rede da próxima geração – IPng.

Com o crescimento da largura de banda [16] vários serviços vão sendo disponibilizados através da Internet, e a Educação à Distância [17], nos meios acadêmicos bem como em processos de treinamentos de funcionários nas empresas, tem se mostrado como uma ferramenta viável, permitindo maior interação dos alunos com o material didático, e com novos modelos de aprendizagem.

Este documento apresenta uma metodologia para aplicação em aulas de laboratório, em um ambiente de sistema operacional aberto, com experimentos didáticos compartilhados, inicialmente em uma rede local, utilizando o protocolo IPv6 [18], e as técnicas desenvolvidas pelo grupo SIM [19] em laboratório Virtual, com o uso do software LabVIEW® da National Instruments [20].

No segundo capítulo são apresentados conceitos sobre Laboratório Virtual, e as tecnologias utilizadas para comunicação através de redes locais e da Internet.

No terceiro capítulo são apresentados conceitos sobre a ferramenta de software LabVIEW®, da National Instruments, descrevendo aplicações básicas, protocolos de comunicação com dispositivos de aquisição de dados, controle de instrumentos, transferência de dados com o DataSocket® e o protocolo TCP/IP.

No quarto capítulo são apresentados conceitos básicos sobre o protocolo IPv6, descrevendo suas características e funcionalidades e comparando com o protocolo IPv4. São apresentados os possíveis cenários de transição para a implementação de sistemas de rede de computadores utilizando o protocolo IPv6.

No quinto capítulo são apresentados os materiais utilizados para construção do Laboratório remoto em ambiente aberto, utilizando o protocolo IPv6. Neste capítulo é apresentada a metodologia utilizada para a implementação do sistema.

No sexto capítulo são apresentados a descrição do experimento utilizado e os resultados experimentais obtidos.

No sétimo capítulo são apresentados a descrição da infra-estrutura de rede, e os resultados experimentais obtidos.

As conclusões do trabalho e perspectivas futuras são descritas no oitavo capítulo.

A bibliografia é apresentada a seguir com as publicações como artigos e livros, além das páginas da Internet (Sites).

2 LABORATÓRIO VIRTUAL

Este capítulo apresenta os conceitos utilizados em sistemas que possibilitam o controle de instrumentação via rede local e através da Internet. A programação em linguagem gráfica, utilizando o paradigma da programação orientada a objeto, implementada no software LabVIEW® da National Instruments, é analisada como uma linguagem de fácil aprendizagem para os usuários, diferente das linguagens de programação em forma de texto. São apresentadas algumas tecnologias que podem ser utilizadas na implementação de Laboratórios Virtuais.

2.1 Instrumentação Virtual

O conceito de instrumentação virtual é uma proposta para controle de processos, inserindo os equipamentos de medida, software e computador, controlando o fluxo de informações entre os dispositivos, utilizando interfaces padronizadas para comunicação IHM – interface homem-máquina. O nome “instrumento virtual” é proveniente dos aspectos realistas de operação dos instrumentos clássicos, como, por exemplo, osciloscópios, amperímetros, voltímetros [21], [22], concebida originalmente como uma ferramenta de desenvolvimento que automatiza os processos industriais e laboratoriais [23].

O ambiente de um Laboratório Virtual é constituído por vários softwares e dispositivos de hardware, sendo responsável pelo controle e acompanhamento remoto do experimento; compatibilidade entre diversas plataformas; interface amigável com o usuário, videoconferência e tratamento de multimídia, gerenciamento e controle do ambiente, permitindo a utilização por profissionais oriundos de outras áreas distantes da engenharia e da computação.

O objetivo de um Laboratório Virtual é criar um ambiente real de colaboração, reunindo diversas tecnologias em uma interface única, por exemplo, um navegador internet – browser. Entre estas tecnologias podem ser citadas: controle de instrumentação, telepresença, gerenciamento e suporte remoto, visualização, distribuição de arquivos e análise de dados on-line e off-line. A Figura 1 mostra um exemplo de instrumentação virtual coletando informações de fenômenos físicos, através de sensores apropriados para a aquisição dos dados, o condicionamento dos

senais, e a apresentação dos resultados, através de vários dispositivos, interfaces e protocolos, com a utilização de software instalado num computador.

Figura 1 – Exemplo de um sistema de instrumentação virtual.¹

2.1.1 Paradigma da programação utilizada no Laboratório Virtual

A evolução do paradigma de programação foi marcada pela transição da linguagem de máquina para as linguagens assembler, a funcional, a procedural e linguagem orientada a objeto. As linguagens de programação procuram resolver os problemas propostos, utilizando uma construção dos programas com uma gramática orientada a texto, combinando uma linguagem próxima da natural e de formalismos matemáticos, dificultando o aprendizado e a sua utilização por profissionais de outras áreas, distantes da engenharia e da computação.

A linguagem de programação gráfica utiliza ícones ao invés de linhas de texto para criar as aplicações. Nas linguagens de programação baseadas em texto, as instruções determinam a execução do programa, enquanto nas linguagens gráficas, o fluxo de dados é quem determina a execução do programa. A programação em linguagem gráfica, normalmente similar a utilização de fluxogramas, tem proporcionado uma condição real na redução do tempo de desenvolvimento da aplicação.

¹ Figura retirada do Seminário NIDays – 2000, promovido pela National Instruments

As interfaces desenvolvidas para o controle através da utilização da instrumentação virtual, segundo Wang [24], devem:

- Executar diferentes testes e funções para a medição de sinais
- Executar o controle do fluxo de informação de dispositivos periféricos (por exemplo, sensores e circuitos de condicionamento/processamento de sinais), utilizando interfaces padrão.
- Adquirir mais de um tipo de dado e mostrar o resultado desta aquisição, acompanhando o fluxo de informação do controle de dispositivos periféricos, possibilitando análises.
- Permitir e facilitar a exibição e análise de dados adquiridos.
- Administrar o gerenciamento de dados, a impressão, a comunicação dos equipamentos conectados em rede, e o acesso à Internet.

2.1.2 Hardware para instrumentação virtual

A medida ou monitoração de dados utilizando computadores consiste de três componentes, cada qual realizando uma função, ou seja, aquisição dos parâmetros de medida (aquisição de dados); adaptação e processamento dos sinais medidos (análise) e a saída do valor medido (apresentação), como mostra a Figura 2.

Figura 2 – Aquisição, análise e apresentação

Os microcomputadores e as técnicas de programação têm proporcionado à instrumentação virtual, formas sofisticadas de controle, e novos padrões de comunicação entre hardware têm sido desenvolvidos [24], oferecendo suporte aos dispositivos e interface para a instrumentação.

2.1.3 Redes de Computadores

A utilização de computadores para a aquisição de dados, e a transferência destes para outros computadores, que podem estar fisicamente próximos ou a milhares de quilômetros de distância, leva a utilização de ambientes de rede. Estes ambientes de interligação tem possibilitado a transferência de informação de forma rápida e segura. Os computadores que utilizamos freqüentemente são chamados de hospedeiros, sistemas finais ou ainda de clientes. São ainda chamados de hospedeiros (“*hosts*”) porque hospedam programas de aplicação [25].

Os sistemas que compartilham a rede são ainda subdivididos em duas categorias: **os clientes e os servidores**, e a ligação entre eles e o compartilhamento de informações segue um modelo chamado **cliente-servidor**. Neste modelo, o cliente possui um programa que pede e recebe informações de um programa servidor que está sendo executado em outro computador.

Este modelo é a estrutura mais utilizada na Internet, onde as aplicações cliente-servidor de Internet são, por definição, **aplicações distribuídas** [25].

A tecnologia de redes se relaciona com a ligação física dos equipamentos, que podem estar geograficamente próximas ou distantes [26]. As redes são divididas em três categorias: as redes de pequena distância ou ainda redes locais (LAN – Local Area Network), as de extensão intermediária – as MANs (Metropolitan Area Network) e as redes de longa distância – as WANs (Wide Area Network) [27]. Neste trabalho estaremos utilizando as redes locais e as redes de longa distância.

A rede local possibilita a conexão de equipamentos onde as interligações dependem de uma infra-estrutura de cabos e conexões, que ficam localizadas dentro das instalações físicas da empresa. Estas redes podem ser construídas com tecnologia de cabos par trançado, fibra óptica ou ainda através de conexões sem fio, dependendo das velocidades desejadas e da estrutura existente no prédio. As redes locais apresentam altas taxas de transferência de dados (que podem variar entre megabits por segundo a gigabits por segundo) e baixo índice de erros e custo relativamente pequeno [26].

As redes remotas, conectando computadores a grande distância, possibilitam a transferência de informações através de um agrupamento de redes locais e redes metropolitanas, construídas em diversas cidades, estados, países e até continentes. As

tecnologias para a construção de redes remotas incluem, cabos da infra-estrutura telefônica, cabos ópticos, ligações submarinas, enlaces de satélite e ainda sistemas terrestres de microondas. A velocidade de transmissão de dados em uma rede WAN sofre influência, principalmente, do número de equipamentos utilizando a rede simultaneamente [26].

A motivação para a utilização de sistemas distribuídos e a Internet ocorreu devido aos avanços na área de infra-estrutura de redes, com novas mídias para a comunicação, principalmente a fibra óptica, e o aumento da largura de banda e com isto, uma melhor qualidade de serviço (QoS) oferecida para as aplicações. Os processos para aquisição de dados podem então ser realizados através da Internet, oferecendo segurança, flexibilidade e maior velocidade no atendimento nos casos de suporte técnico às instalações insalubres [28].

2.1.4 Internet e os serviços disponíveis

A divulgação da ciência e o conhecimento para a comunidade científica/acadêmica, as indústrias, as empresas e a sociedade, de um modo geral, nos levam cada vez mais a utilizar a Internet como uma ferramenta poderosa, que permite de forma simples e fácil, a conexão em uma grande “Rede de Serviços”, onde diversos protocolos e tecnologias trabalham de forma cooperada.

Nessa seção vamos descrever alguns protocolos utilizados na camada de aplicação do conjunto de protocolos TCP/IP, mostrados na Figura 3.

Figura 3 – Protocolos utilizados nas diversas camadas do TCP/IP

2.1.4.1 O protocolo FTP

O protocolo FTP (“File Transfer Protocol”) permite ao usuário obter arquivos de outro computador, ou ainda, enviar arquivos para outro computador, definido como servidor de FTP. As senhas de proteção são incorporadas como parte do protocolo, e ao conectar-se no computador servidor de FTP, serão solicitados um nome de usuário e uma senha. Geralmente para transferência de arquivos é utilizado o “*anonymous*” FTP.

O protocolo de transferência de arquivos envolve duas conexões diferentes. Na primeira, o programa do usuário envia o comando "conecte-me como este usuário e com esta senha, e envie-me o arquivo com este nome".

Assim que o comando para mandar dados é enviado, uma segunda conexão é aberta para seus dados.

Certamente seria possível enviar dados em uma conexão, como no caso de correio eletrônico, entretanto, transferências de arquivos geralmente levam algum tempo, e os projetistas do FTP desejavam permitir que o usuário continuasse emitindo comandos, enquanto a transferência era feita, por exemplo, solicitar uma informação ou abortar a transferência.

O serviço de FTP possui as seguintes facilidades:

- **Acesso interativo:** embora o FTP tenha sido projetado para ser usado por programas, a maioria das implementações apresenta uma interface interativa, o que torna a interação com servidores remotos mais fáceis.
- **A especificação do formato FTP:** permite ao cliente definir o tipo e o formato dos dados armazenados.
- **Controle de autenticação FTP:** solicita aos clientes a sua autorização para enviar um nome de “*login*” e uma senha, para o servidor, antes de solicitar a transferência de arquivo. O servidor recusa o acesso do cliente, caso o “*login*” e a senha, não forem válidos.

2.1.4.2 O protocolo TELNET

O protocolo TELNET emula um terminal, possibilitando ao usuário, o acesso de aplicações em outros computadores, de outras redes.

A sessão remota é iniciada especificando-se em qual computador e rede, o usuário deseja conectar-se. Com a conexão remota estabelecida, uma sessão remota é aberta, e qualquer coisa que é digitada no computador local, é enviada diretamente para o computador remoto. Para a conexão remota, também serão solicitados um nome de usuário e uma senha.

O protocolo TELNET oferece três serviços básicos:

- Define um terminal virtual de rede, com uma interface padrão para sistemas remotos; os programas clientes não têm que compreender os detalhes de todos os possíveis sistemas remotos, sendo feitos para usar a interface padrão.
- Mecanismo que permite ao cliente e ao servidor negociarem opções e proporcionar um conjunto de opções padrão para a comunicação.
- Tratamento das pontas da conexão simetricamente. Assim, ao invés de forçar o cliente para conectar-se a um terminal de usuário, o TELNET permite um programa arbitrário tornar-se um cliente.

2.1.4.3 O protocolo SMTP

O protocolo SMTP ("Simple Mail Transfer Protocol" - protocolo de transferência de correio simples), possibilita o envio, recebimento e armazenamento de mensagens eletrônicas, para usuários de outros computadores, observando-se os endereços eletrônicos.

O programa de correio eletrônico inicia a chamada ao servidor de correio, que está na outra ponta da conexão, enviando o nome da máquina, o remetente e o conteúdo da mensagem. O servidor de correio aceita a conexão, sinalizando para a ponta remetente, que começa então a enviar o texto da mensagem.

2.1.4.4 O protocolo HTTP

A Internet é uma aplicação de rede e possibilita que os usuários tenham acesso a documentos de servidores WEB; é composta de muitos componentes e entre eles existe um padrão para o formato de documentos: a linguagem HTML – HyperText Markup Language, os navegadores – “browsers” WEB, os servidores WEB e o protocolo da camada de aplicação – o HTTP – Hypertext Transfer Protocol (protocolo de transferência de hipertexto). Este protocolo define como as mensagens são transferidas entre o “browser” do cliente e o servidor WEB, sendo apenas um pedaço da aplicação WEB.

Uma página WEB, também chamada de documento, possui diversos objetos, que podem ser um arquivo escrito em linguagem HTML, uma imagem no formato JPEG (Joint Photographic Experts Group) ou no formato GIF (Graphics Interchange Format), um arquivo de vídeo, um arquivo de áudio, um aplicativo escrito na linguagem Java, e que podem ser acessados por meio de um URL - Uniform Resource Locator, que nos dá o endereço de acesso a um recurso na Internet. [25].

A maioria das páginas WEB é composta de um arquivo-base HTML e objetos relacionados. Cada URL tem dois componentes: o nome do hospedeiro (host) do servidor, onde está armazenado o objeto e o nome do caminho do objeto, por exemplo: www.lme.usp.br/sim/labvirtual.gif, que tem www.lme.usp.br como nome de hospedeiro e [/sim/labvirtual.gif](http://www.lme.usp.br/sim/labvirtual.gif) como nome do caminho para o objeto, que no caso é o arquivo de imagem [labvirtual.gif](http://www.lme.usp.br/sim/labvirtual.gif).

O protocolo HTTP está implementado em dois programas: um programa cliente e um programa servidor, que são executados em sistemas diferentes, que interagem trocando mensagens. O protocolo HTTP define como os clientes solicitam páginas WEB aos servidores WEB. Quando um usuário solicita uma página, através de um endereço URL, o “browser” envia ao servidor mensagens de requisição HTTP para os objetos da página. O servidor recebe as requisições e responde com uma mensagem de resposta HTTP que contém os objetos. A Figura 4 mostra o comportamento de uma requisição e resposta do protocolo HTTP.

Figura 4 – Comportamento de requisição-resposta do protocolo HTTP

O protocolo HTTP utiliza o protocolo TCP para a camada de transporte. O cliente HTTP inicia uma conexão TCP com o servidor, e ao ser estabelecida, os processos do “browser” e do servidor acessam o TCP por meio de sua interface de porta. A porta de acesso ao serviço HTTP é a porta 80, que é utilizada como porta padrão.

2.1.5 Sistemas Cliente/Servidor

Os laboratórios virtuais utilizam arquiteturas de comunicação baseadas na tecnologia cliente/servidor, onde dois ou mais computadores trabalham para o processamento dos dados, sendo divididos em módulos denominados Servidor e Cliente.

A aplicação cliente possibilita aos usuários o acesso a serviços que são disponibilizados pelo servidor. As solicitações são processadas pelo servidor e depois envia os resultados aos clientes que fizeram as solicitações.

A arquitetura cliente/servidor apresenta as vantagens descritas a seguir:

- **Escalabilidade** – o sistema cliente/servidor pode ser expandido verticalmente, adicionando-se recursos ao computador.

- **Flexibilidade** – vários clientes podem acessar os recursos, de diferentes pontos da rede.
- **Independência de plataforma** – os sistemas cliente/servidor não dependem de um ambiente de *hardware* ou *software* específico.
- **Facilidade de acesso** – o servidor tem mais liberdade para manipulação das informações, pois o processo cliente é que gerencia a interface.
- **Redução de custos operacionais** – a atualização dos sistemas pode ser realizada com baixo custo.

Os sistemas de controle utilizando a Internet podem ser implementados através da arquitetura cliente/servidor, devendo o servidor estar instalado próximo dos instrumentos e do experimento, e então transferir os comandos ao cliente [29].

A Figura 5 mostra uma arquitetura de rede cliente/servidor utilizada para controle dos instrumentos.

Figura 5 – Arquitetura de uma rede cliente/servidor e o controle dos instrumentos

2.1.6 Considerações finais

Neste capítulo foram apresentados conceitos sobre sistemas de controle de instrumentação via rede local e através da Internet, bem como a descrição de algumas tecnologias que possibilitam a implementação de sistemas de aquisição, controle e apresentação de dados.

Como conclusão temos a descrição de uma estrutura básica para a implementação de um laboratório virtual, utilizando a tecnologia cliente/servidor.

O crescimento das redes de computadores, os protocolos de comunicação e as ferramentas de desenvolvimento com características de programação gráfica, contribuem tecnologicamente com a motivação inicial para o desenvolvimento de um Laboratório Virtual, e com isto o compartilhamento de aplicações nas mais diversas áreas do conhecimento.

3 A FERRAMENTA DE SOFTWARE - LABVIEW®

Neste capítulo serão apresentadas as características da ferramenta LabVIEW®, utilizada na implementação do sistema de aquisição, controle e apresentação do Laboratório Virtual. A estruturas de programação, os protocolos utilizados para aquisição de dados e comunicação com os instrumentos e os protocolos de transferência de dados em redes de computadores, também são analisados.

3.1 Características básicas do LabVIEW®

O sistema de instrumentação virtual é descrito através de quatro parâmetros: linguagem de programação, estrutura de controle, latência e os dispositivos específicos [30]. A linguagem de programação utilizada é a gráfica; a estrutura de controle utiliza diagramas de blocos executando seqüências lógicas para controlar um processo; a latência, que é um a parâmetro ligado ao tempo de resposta entre a mensagem do software e os instrumentos, e os dispositivos específicos, inseridos para uma determinada aplicação.

No software LabVIEW® da National Instruments, a interface de usuário é construída com a utilização de um conjunto de ferramentas e objetos, sendo conhecida como painel frontal. O diagrama de blocos contém os códigos do projeto, com a inserção de representações gráficas das funções necessárias para controlar os objetos colocados no painel frontal.

Os programas em LabVIEW® são chamados de *instrumentos virtuais*, ou VIs, devido sua aparência e modo de operação, semelhante aos instrumentos físicos, como osciloscópios e multímetros. Cada VI utiliza funções para manipular as entradas, vindas de interfaces de usuário ou outras fontes e apresenta a informação ou armazena em arquivos ou até mesmo em outros computadores.

A programação em LabVIEW® utiliza a linguagem gráfica, para a construção do instrumento virtual – VI, que contém três componentes:

- **Painel frontal** – utilizado como interface de usuário.

- **Diagrama de blocos** – contém o código fonte que define a funcionalidade do instrumento virtual.

- **Ícone e painel de ligação** - identifica um instrumento virtual, que pode utilizar um outro VI, sendo esta característica chamada de sub VI. Uma sub VI corresponde a uma sub-rotina das linguagens de programação baseada em texto.

O painel frontal é o VI de interface de usuário, construído com controles e indicadores, os quais interagem com os terminais de entrada e saída do instrumento virtual, respectivamente. Os controles de entrada são botões, chaves rotativas e outros dispositivos de entrada. Os indicadores são as janelas oscilográficas, os indicadores luminosos (“LEDs”), e outros dispositivos apropriados para mostrar os dados na saída. Os controles simulam os dispositivos de entrada dos instrumentos e fornecem os dados para o diagrama de blocos do VI. Os indicadores simulam os dispositivos de saída e apresentam os dados que o diagrama de blocos recebeu ou gerou.

Na Figura 6 temos o painel frontal inicial para a construção de uma aplicação utilizando o software LabVIEW®.

Figura 6 – Painel frontal de uma aplicação em LabVIEW®

Depois da construção do painel frontal, podemos adicionar código utilizando a linguagem de programação gráfica, representando as funções para controlar os objetos colocados no painel frontal. O diagrama de blocos contém o código gráfico, fonte da aplicação. Os objetos do painel frontal aparecem como terminais no

diagrama de blocos. O diagrama de blocos pode conter também funções e estruturas construídas pelo LabVIEW® e armazenadas como bibliotecas de VIs, o que acelera o desenvolvimento da aplicação. As ligações interligam cada um dos nós no diagrama de blocos, incluindo os controles e os terminais indicadores, as funções e as estruturas.

O LabVIEW® utiliza um modelo de fluxo de dados para executar o VI. O nó do diagrama de blocos, só é executado quando todas as suas entradas estão disponíveis. Quando um nó, completa a execução, ele fornece os dados para seus terminais de saída, passando-os para o próximo nó no caminho do fluxo de dados.

Na Figura 7, temos o diagrama de blocos inicial para a construção de uma aplicação, utilizando o software LabVIEW®.

Figura 7 – Diagrama de blocos de uma aplicação em LabVIEW®

3.2 Estruturas de programação em LabVIEW®

O LabVIEW® utiliza a linguagem de programação gráfica e através dela, com os componentes visuais, é expressa a lógica do programa. Os componentes também representam os blocos construtivos de uma linguagem de programação: blocos iterativos (for, while), estruturas de decisão (case, if-then-else), sub-rotinas e procedimentos, estruturas de dados (as variáveis, cadeias de caracteres, matrizes, estruturas híbridas). Estas estruturas são utilizadas no diagrama de blocos para repetir blocos de código e para executar condicionalmente, ou em uma sequência específica.

Assim como outros os nós, as estruturas possuem terminais que são conectados a outros nós do diagrama de blocos, executados automaticamente quando os dados de entradas estiverem disponibilizados, e fornecendo dados para as saídas, quando completada a execução.

Os blocos iterativos implementados no LabVIEW[®] podem executar um sub-diagrama por um número de vezes (*“loop for”*), mostrado na Figura 8.

Figura 8 – Bloco iterativo for

O valor N colocado no terminal de entrada indica quantas vezes será repetido o bloco e o terminal de saída indica o número de iterações completadas, sendo que durante a primeira iteração seu valor será zero. Estes parâmetros são valores inteiros.

A Figura 9 mostra a estrutura de repetição que é executada até que uma condição de teste seja atingida (*“loop while”*). O terminal de entrada de condição recebe um valor booleano, sendo que a condição padrão é que o *“loop”* continuará sendo executado enquanto a condição booleana for verdadeira. Esta configuração pode ser parametrizada conforme a necessidade do projeto. O terminal de saída do *“loop”* indica o número de iterações completado.

Figura 9 – Bloco iterativo “while”

A Figura 10 mostra uma estrutura que contém vários sub-diagramas, onde apenas um deles é executado, dependendo do valor de entrada fornecido para a estrutura. Esta estrutura tem dois ou mais diagramas, sendo que apenas um diagrama será visível em um momento e a estrutura executará apenas um caso, sendo que o

valor de entrada determinará qual o caso será executado. Esta estrutura é similar ao “case” ou o “if... then... else” das linguagens de programação baseadas em texto.

Figura 10 – Estrutura do bloco “case” nas condições True e False

A Figura 11 mostra uma estrutura de sequência que contém um ou mais sub-diagramas, os quais são executados em uma ordem seqüencial.

Figura 11 – Bloco de estrutura de sequência

A Figura 12 mostra um exemplo de uma aplicação utilizando as estruturas de repetição “for” e a estrutura de sequência.

Figura 12 – Exemplo de um diagrama utilizando as estruturas de repetição e de sequência.

3.3 Interfaces de aquisição de dados

Para a aquisição de informação, análise e monitoração através de sistemas computadorizados, a interface entre um computador e seu usuário ocorre através de dispositivos de entrada/saída: o teclado, a tela do monitor de vídeo, o apontador (*mouse*, “*trackball*”) e vários outros dispositivos que surgem para facilitar a interface homem-máquina. A interligação com outros computadores e outros usuários, compartilhando informações e periféricos, leva a utilização das interfaces de rede, permitindo iterações remotas e transferências de informações. Para a comunicação com dispositivos periféricos ligados ao computador para a aquisição e monitoração, são utilizados adaptadores específicos para cada aplicação (porta serial, porta paralela, placas dedicadas). Estes equipamentos são, por exemplo, instrumentos de medida, câmeras de vídeo, sensores, motores, etc.

O computador é utilizado tanto para operar o instrumento, quanto para conduzir o experimento, e nessa situação, a interface associada ao computador deve proporcionar a mesma funcionalidade que o equipamento real apresentaria ao usuário, estando limitada às características da ferramenta de software de instrumentação utilizado.

Para a conexão de instrumentos reais com um computador, deve ser considerado o componente de interfaceamento, que se refere ao módulo de software, representando o instrumento dentro do ambiente de programação e o padrão de comunicação, que determina a forma como os dados são transferidos.

Os tipos de barramentos de dados mais comuns para uso com um microcomputador são as portas seriais e a paralela. No barramento serial, existe uma única via de comunicação, por onde os bits de dados passam serialmente. No barramento paralelo, existem várias vias para a transmissão dos dados, paralelamente. Existem normas que padronizam os parâmetros dessas vias e a forma como são inseridos os dados e sinais nas vias, chamados de protocolo. Os protocolos seriais mais utilizados são o RS-232C, o RS-449 A, o Universal Serial Bus (USB) e o IEEE-1394 (FireWire). Os protocolos paralelos mais usuais são o IEEE-1284 (Centronics) e o IEEE-488 (GPIB).

3.3.1 Comunicação Serial

A comunicação serial é o mais popular meio de transmissão de dados entre um computador e um dispositivo periférico, por exemplo, um instrumento programável ou ainda um outro computador. A comunicação serial utiliza um transmissor para enviar os dados, um bit no tempo, sobre uma linha simples de comunicação. Este método pode ser utilizado quando as taxas de transferência de dados são baixas, ou ainda quando as distâncias são muito longas. A comunicação serial é muito popular pelo fato dos computadores possuírem uma ou mais portas seriais, não necessitando hardware extra, além de um cabo para conectar o instrumento a um computador, ou interligar dois computadores. A Figura 13 mostra a comunicação entre um microcomputador e um instrumento programável, com interface de comunicação serial.

Figura 13 – Comunicação serial entre microcomputador e um instrumento

A comunicação serial requer que sejam especificados quatro parâmetros: “*baud rate*” da transmissão, o número de bits de dados codificados como caractere, o bit de paridade (opcional) e o número de bits de parada (“*stop bits*”). Cada caractere transmitido é empacotado em um conjunto de bits que inclui um bit sinalizador de início, os bits de dados, o bit de paridade e os bits de término (“*stop bits*”). A Figura 14 mostra o caractere “m” codificado para a transmissão.

Figura 14 – Transmissão serial do caractere “m”

A comunicação serial utiliza apenas dois níveis de tensão, denominados MARK e SPACE, definidos como tensões negativa e positiva, respectivamente. Os níveis de tensão representam os estados lógicos zero (sinal maior que +3 V) e um (sinal menor que -3 V). O sinal de saída pode estar entre os níveis de tensão +12 V e -12 V. A região entre os níveis de tensão +3 V e -3 V é utilizada para absorver os ruídos na linha de comunicação.

Para identificar o início da transmissão de um novo caractere, é inserido um bit, representado pela transição de uma tensão negativa (“MARK”) para uma positiva (“SPACE”). Os bits representando os dados são transmitidos na ordem do bit menos significativo para o mais significativo. Para identificar o código do caractere mostrado na Figura 14 deve-se ler o conjunto de bits, após o bit de início, direita para a esquerda, assumindo nível lógico um para as tensões negativas e nível lógico zero para as tensões positivas, o que dá a representação binária 1101101 ou ainda o valor 6D na base hexadecimal. A tabela de conversão para ASCII (American Standard Code for Information Interchange) mostrada na Figura 15 indica que o caractere é a letra “m”.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NUL	SOH	STX	ETX	END	ACK	BEL	BS	HT	LF	VT	FF	CR	SO	SI	
1	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	BS	RS	US
2	SPC	!	"	#	\$	%	&	'	(	)	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{		}	~	DEL

Figura 15 – Tabela de caracteres ASCII

O bit de paridade pode ser colocado após os bits referentes ao código do caractere, sendo utilizado para tratamento de erros nos bits de dados. A paridade pode ser definida como par ou ímpar. Se a paridade escolhida é ímpar, o transmissor coloca o bit de paridade num valor tal, que em conjunto com os bits de dados, totalize um número ímpar de bits em nível um.

Os últimos bits colocados no conjunto de bits, na transmissão de um caractere, são utilizados como marcação de fim de caractere, podendo ser um, um e meio, ou ainda dois bits, sendo representados por uma tensão negativa. O próximo caractere a ser transmitido é então sinalizado pela transição da tensão para um nível positivo, representando o bit de início.

Existem muitos padrões recomendados para portas de comunicação seriais, e entre eles temos o RS-232, e os padrões RS-422 e RS-485, adicionais ao RS-232. O padrão RS-232 foi adotado para comunicação serial entre computadores e os equipamentos, como impressoras, modems e outros dispositivos. As portas seriais para os computadores compatíveis com o PC-IBM, utilizam o padrão RS-232, com a identificação dos terminais mostrada na Figura 16.

Figura 16 – Porta serial RS-232 de nove pinos

A Tabela 1 mostra os terminais do conector de nove pinos, utilizado na interface serial de micros padrão PC-IBM.

Tabela 1 – Terminais do conector serial de um microcomputador padrão PC-IBM

Pino	Nome	Tipo	Função
1	Carrier Detect	Entrada	Determina se a porta está conectada para trabalhar com o dispositivo.
2	Receive Data	Dado	Recebe informação de um dispositivo.
3	Transmit Data	Dado	Computador envia informação para o dispositivo.
4	Data Terminal Ready	Saída	Computador informa ao dispositivo que está pronto para comunicação.
5	Signal Ground	Comum	Terminal terra.
6	Data Set Ready	Entrada	Dispositivo informa ao computador que está pronto para comunicação.
7	Request To Send	Saída	Computador questiona o dispositivo se ele pode enviar informação.
8	Clear To Send	Entrada	Dispositivo informa ao computador que pode enviar informação.
9	Signal Indicator	Entrada	Computador reconhece que um sinal foi detectado.

3.3.2 Arquitetura de Software de Instrumentos Virtuais -VISA

A Arquitetura de Software de Instrumentos Virtuais (VISA) é a camada inferior das funções do software do VI – instrumento virtual – que comunica com os dispositivos, e foi desenvolvida para ter as seguintes propriedades:

- Maximizar a facilidade de uso e performance.
- Manter a compatibilidade com a base instalada.
- Manter arquiteturas abertas.
- Maximizar a capacidade de ser multiplataforma.
- Maximizar a expansibilidade e a modularidade.
- Maximizar a reutilização de software.
- Padronizar o uso de objetos nos sistemas de software.
- Tratar os “*drivers*” de instrumentos como parte do instrumento.
- Acomodar e estabelecer padrões.
- Maximizar o suporte cooperativo entre os usuários.

VISA não proporciona, por si própria, capacidades de programação, mas é uma Interface de Programação de Aplicação (API) que conecta aos “*drivers*”, podendo controlar dispositivos VXI (VMEbus eXtensions for Instrumentation), GPIB (General Purpose Interface Bus), seriais ou instrumentos baseados em computador e tornar a chamada ao “*driver*” dependente do tipo de instrumento utilizado, tornando mais rápido o desenvolvimento do software. A arquitetura VISA é mostrada na Figura 17. No LabVIEW®, VISA é uma biblioteca de funções, que são utilizadas para comunicação com os instrumentos.

Figura 17 – Arquitetura VISA e os dispositivos de comunicação

As funções utilizadas com o recurso são as operações. Os recursos também possuem variáveis, ou atributos e informações relacionadas. A terminologia é similar à utilizada para o VI – instrumento virtual:

- **Recurso** – um instrumento do sistema, incluindo portas seriais e paralelas.
- **Sessão** – abrir uma sessão VISA para que o recurso possa comunicar com ele, similar a um canal de comunicação. Quando uma sessão é aberta, o LabVIEW® retorna um número de sessão VISA, único para o instrumento.
- **Descritor do instrumento** – nome exato do recurso. O descritor especifica o tipo de interface (GPIB, VXI, ASRL), o endereço do dispositivo (endereço lógico ou endereço primário), e o tipo de sessão VISA (INSTR ou Evento).

O descritor do instrumento é similar a um número de telefone, o recurso é comparado a uma pessoa com quem desejamos falar, e a sessão é similar com a linha telefônica. Assim, cada chamada utiliza uma linha própria, e o cruzamento delas, resulta em erro.

A função VISA Write, mostrada na Figura 18, escreve o string **write buffer** no dispositivo especificado pelo nome **VISA resource name**. O **dup VISA resource name** retorna o mesmo identificador para a sessão. Nas plataformas UNIX, os dados são escritos de forma síncrona. Em todas as outras plataformas, os dados são escritos de forma assíncrona. O terminal **return count** contém o número de bytes atualmente transferidos através do GPIB. Os terminais **error in** e **error out** contém informações de erros.

Figura 18 – VISA Write: escrever dado em um dispositivo especificado

A função VISA Read, mostrada na Figura 19, lê os dados vindos de um dispositivo especificado pelo terminal **VISA resource name**. O terminal **byte count** indica o número de bytes a serem lidos através do terminal **read buffer**. O terminal **dup VISA resource name**, retorna o mesmo identificador para a sessão. Nas

plataformas UNIX, os dados são escritos de forma síncrona. Em todas as outras plataformas, os dados são escritos de forma assíncrona. O terminal **return count** contém o número de bytes atualmente transferidos através do GPIB. Os terminais **error in** e **error out** contém informações de erros.

Figura 19 - VISA Read: lê dados de um dispositivo especificado.

Na Figura 20 temos a implementação de um VI, mostrando a utilização dos blocos de funções VISA, onde a porta serial é configurada e preparada para a transmissão, um conjunto de dados é escrito e convertido para o formato apropriado, incluindo os bits de início, de paridade e de parada.

Figura 20 – Diagrama de blocos e painel frontal para um VI de teste de loopback serial

Na Figura 21 temos a implementação de um VI, representando um instrumento utilizado para a simulação de escrita e leitura de dados através da porta de comunicação de um computador.

Figura 21 – Diagrama de blocos e painel frontal para um VI de um Simulador de escrita e leitura

AS FUNÇÕES NI-VISA PROPORCIONAM UMA MANEIRA FÁCIL DE IMPLEMENTAÇÃO DE COMUNICAÇÃO COM INSTRUMENTOS UTILIZANDO DIFERENTES PROTOCOLOS, COMO, POR EXEMPLO, SERIAL E GPIB (GENERAL PURPOSE INTERFACE BUS).

3.3.3 Barramento de Interface de uso geral - GPIB

O padrão GPIB usualmente chamado de IEEE 488 e HPIB, utiliza uma interface e barramento para a comunicação entre um computador e um instrumento (osciloscópio, multímetro, etc.), e o cabo GPIB, mostrado na Figura 22.

Figura 22 – Placa PCI-GPIB com cabo da National Instruments

O controlador pode especificar um dispositivo individual, entretanto usualmente se refere à placa PCI-GPIB ou uma interface similar conectada num computador, possibilitando ao usuário interagir com o instrumento. A Figura 23 mostra um sistema com a interface GPIB, onde os usuários podem utilizar a versatilidade dos instrumentos e dos computadores, na construção de aplicações.

Figura 23 – Comunicação entre instrumentos e computador utilizando GPIB

O padrão GPIB foi criado para proporcionar a comunicação e o controle de múltiplos instrumentos, facilmente adaptado para controle através de um computador, com velocidades de transferência de dados alta, permitindo o envio de grandes quantidades de informação. Muitos instrumentos, atualmente, já possuem suporte ao protocolo GPIB. Com o padrão GPIB é relativamente fácil programar e estabelecer a comunicação com um instrumento [31].

O GPIB é uma interface de comunicação paralela de oito bits, digital, com taxas de transferência acima de um Mbyte/s, utilizando uma conexão com três fios. O barramento suporta um Sistema Controlador, geralmente um computador e quatorze instrumentos adicionais [32], que podem ser interligados em uma configuração estrela ou em uma configuração linear, mostradas na Figura 24.

Figura 24 – Configuração estrela e configuração linear de equipamentos com GPIB

A Figura 25 mostra o conector padrão GPIB de 24 pinos, com oito linhas de dados, cinco linhas de gerenciamento do barramento (ATN, EOI, IFC, REN, e SRQ), três linhas de controle e oito linhas de aterramento. O GPIB utiliza a comunicação serial assíncrona para a transferência de bytes, o que faz com que a transmissão dos bits através do barramento seja determinada pelo dispositivo mais lento. As mensagens são caracteres codificados com o código ASCII.

Figura 25 – Especificações do conector GPIB

Os dispositivos GPIB comunicam-se uns com os outros enviando dois tipos de mensagens:

- **Independentes do dispositivo:** são os dados, que podem ser instruções de programação, situação da máquina, resultados de medidas, ou ainda arquivos de dados;
- **Comando de interface:** destinadas à execução de funções como, por exemplo, endereçamento de dispositivos, programação da interface, inicialização do barramento da interface, etc.

O padrão GPIB prevê três tipos de dispositivos com atividades específicas. Existem os que falam (“*talkers*”) – enviam mensagens de dados, os que escutam (“*listeners*”) – recebem mensagens de dados e o controlador - que gerencia o fluxo de informações no barramento, sendo normalmente um computador. Ele define os canais de comunicação e envia comandos GPIB para os dispositivos. Os instrumentos virtuais GPIB manuseiam automaticamente o endereçamento e outras funções de gerenciamento do barramento.

O padrão GPIB apresenta as seguintes restrições de montagem:

- Separação máxima entre dois dispositivos: 4 m
- Separação média entre dispositivos: 2 m
- Máximo comprimento do cabo: 20 m
- Número máximo de instrumentos em cada barramento: 15

O LabVIEW[®] pode se comunicar com instrumentos GPIB através de uma das duas possibilidades:

- O VI do instrumento de Entrada/Saída e as funções comunicam com o “*driver*” de software para a interface GPIB, o que permite construir VIs que utilizam o VI do instrumento diretamente. Entretanto, os instrumentos possuem seus conjuntos de comandos e protocolos para envio e recepção de dados, o que pode dificultar o aprendizado.

- Um “*driver*” de instrumento é um conjunto modular de funções de software, que utiliza os comandos ou protocolo para realizar as operações com o instrumento. O “*driver*” de instrumento também invoca VIs apropriados e as funções para o instrumento. Os “*drivers*” de instrumentos do LabVIEW[®] eliminam a necessidade de se aprender a complexidade de programação de cada instrumento.

Os “*drivers*” de instrumento recebem, analisam e colocam em sequência os caracteres de resposta vindos dos instrumentos, para serem utilizados nas aplicações de teste, e a manutenção da aplicação se torna mais fácil, pois os “*drivers*” armazenam todas as entradas/saídas numa biblioteca separada do código da aplicação. Quando ocorre uma atualização no hardware, a aplicação é atualizada de forma mais fácil, pois todo o código específico para o instrumento, está contido no “*driver*” do instrumento.

As funções de alto nível, mais fáceis de se trabalhar, são construídas a partir das funções de baixo nível, que são utilizadas para um maior controle do instrumento.

A Figura 26 mostra a hierarquia típica de um “*driver*” de instrumento, separada nas categorias: inicialização, configuração, ação e “status”, dados, utilitários e fechar, que têm suas funcionalidades descritas a seguir:

Figura 26 – Hierarquia típica de um *driver* de instrumento

- **Inicialização** – inicializa o canal de comunicação com o instrumento. Este VI pode realizar um teste de identificação e uma operação de reinicialização, colocando o instrumento num estado padrão após ter sido ligado, ou ainda num estado específico.

- **Configuração** – configura o instrumento para realizar operações, por exemplo, fixar uma velocidade de disparo.

- **Ação/Estado** – contém dois tipos de VIs, sendo que a *VI Ação* força o instrumento a iniciar ou terminar as operações de teste e medidas. A *VI estado* obtém o estado atual do instrumento ou o estado de operações pendentes.

- **Dado** – transfere dado para ou do instrumento, por exemplo, a leitura de uma medida de uma forma de onda ou o envio de um sinal para o instrumento.

- **Utilitário** – realiza uma grande variedade de funções, como reinicialização, chamada de testes de erros e testes de revisão.

- **Fecha** – fecha o canal de comunicação para o instrumento, liberando os recursos alocados anteriormente para o instrumento.

Quando um canal de comunicação com um instrumento é inicializado, deve-se conhecer o nome do recurso ou o descritor do instrumento. Um recurso é um instrumento ou interface, e o descritor do instrumento é o nome exato e a localização, descrita no formato: *Interface Type[board index]:: Address::INSTR*

Por exemplo: GPIB:: 2::INSTR é um descritor de um instrumento para um instrumento GPIB no endereço 2.

Após a inicialização de um instrumento, é retornado um número de sessão VISA, que é a conexão ou canal para o instrumento. Cada vez que for utilizado o dispositivo, deve ser ligado este número de sessão com o VI do instrumento. Após o término da comunicação com o instrumento, deve-se utilizar o *VI Fechar*, para fechar todas as referências ou recursos para o instrumento. As funções utilizadas para a comunicação com um instrumento GPIB são a VISA Write e a VISA Read, para escrita e leitura de dados, respectivamente.

Como exemplo, construímos um VI que será usado para mostrar o histórico de uma tensão de um sinal randômico, e registrar as tensões máxima e mínima, e também mostrar como fixar uma faixa de tensão inicial.

A Figura 27 mostra o painel frontal para um VI Histórico de Tensão, com um indicador gráfico registrando os pontos, indicador de tensões máxima e mínima e possibilita o ajuste da faixa de tensão, além da configuração do canal GPIB.

Figura 27 – Painel frontal para o VI Histórico de tensão

A Figura 28 mostra o diagrama de blocos utilizado para a implementação do VI Histórico de tensão.

Figura 28 – Diagrama de blocos para o VI Histórico de tensão

3.3.4 Aquisição de dados – DAQ

O interfaceamento com dispositivos sensores e atuadores que trabalham diretamente com sinais analógicos, pode ser conseguido com a utilização de dispositivos de aquisição de dados (DAQ – Data Acquisition), que possuem conversores analógico-digitais, e são interligados diretamente ao barramento de um computador, ou ainda, com a utilização de dispositivos externos, conectados via comunicação serial, paralela, TCP/IP e outras. Os dispositivos DAQ, da National

Instruments, incluem utilitários para configuração e testes, sendo chamados de Propriedades e Pannel de testes.

Uma placa de aquisição de dados – DAQ- deve ser configurada para ligação em um computador, e os parâmetros necessários são:

- **Endereço Base de E/S** – a comunicação de um dispositivo DAQ com o computador ocorre através de registros. A placa NI-DAQ possui registros de configuração do dispositivo, que são escritos inicialmente, e os dados são lidos depois nos registros de dados do dispositivo, para obter o estado ou um sinal medido. O endereço base de E/S determina onde, no espaço de endereçamento de dispositivos de entrada/saída do computador, os registros do dispositivo serão colocados.

- **Requisição de Interrupção (IRQ)** – um outro modo do dispositivo DAQ se comunicar com o computador, é através das interrupções do processador, as quais permitem ao processador responder às solicitações de periféricos. No caso de dispositivos DAQ, não é um método eficiente, pois forçaria o processador a estar continuamente monitorando se um dado está pronto para ser lido do dispositivo. A utilização do dispositivo DAQ é sinalizar ao processador que existe um dado esperando para ser lido.

- **Acesso Direto a Memória (DMA)** – este é o terceiro modo que o dispositivo DAQ pode se comunicar com o processador, através do acesso direto à memória. Este método permite a transferência de dados entre o dispositivo e a memória do computador, sem a interferência do processador. É requerido quando se deseja a máxima velocidade de transferência de dados, nos dispositivos DAQ de alta velocidade. Os dispositivos que utilizam o barramento PCI possuem seus próprios canais de DMA.

- **AI** - configura a polaridade/faixa padrão do conversor analógico-digital - ADC e o modo padrão de aterramento para o dispositivo DAQ.

- AO – configura a polaridade padrão para o sinal analógico de saída, e possibilita especificar se a tensão de referência que está sendo utilizada pelo conversor analógico-digital é externa.

A Figura 29 mostra um VI que mede a tensão de um sensor de temperatura, e retorna um valor de tensão proporcional à temperatura. O sensor de temperatura está conectado no canal zero da placa de aquisição de dados.

Figura 29 – Painel frontal de um instrumento voltímetro

A Figura 30 mostra o diagrama de blocos construído para o VI voltímetro, utilizado para a medida de tensão.

Figura 30 – Diagrama de blocos do VI do voltímetro

Para uma aquisição com “buffer”, o LabVIEW[®] informa ao dispositivo DAQ quantos pontos serão lidos e a velocidade de aquisição, sendo que o dispositivo controla todos os aspectos da aquisição. Nas aquisições de dados, onde o software controla o tempo de aquisição, o computador é o único responsável pelo

gerenciamento da aquisição, e isto pode ser problemático se o computador não puder dar prioridade para o processo de aquisição de dados.

A Figura 31 e a Figura 32, respectivamente, mostram a o painel e o diagrama de blocos para um instrumento virtual com a utilização do “buffer” para controlar o tamanho da amostra a ser recebida e a taxa de varredura (“scan”).

Figura 31 – Painel frontal do VI utilizando controle de buffer e taxa de varredura

Figura 32 – Diagrama de blocos do VI utilizando controle de buffer e taxa de varredura

3.4 Protocolos de comunicação

Nesta seção vamos analisar os protocolos DataSocket® e TCP/IP, utilizados pelo software LabVIEW®, para disponibilizar o acesso e a transferência de dados numa rede local ou através da Internet.

3.4.1 Protocolo DataSocket®

O DataSocket® é uma tecnologia de programação, baseada no padrão TCP/IP, que simplifica a troca de dados entre diferentes aplicações num computador, ou entre computadores em uma rede. O DataSocket® implementa facilidade de utilização, e uma interface de programação de alta performance, projetada para compartilhar e publicar dados em aplicações de automação. [33]

Com o DataSocket® podemos especificar as fontes de dados e os destinatários para as conexões utilizando URL's (Uniform Resources Locators), como por exemplo, `http://...`

O DataSocket® é composto de duas partes: o DataSocket® API – que apresenta uma interface simples para comunicação com vários tipos de dados, vindos de diversas linguagens de programação, e o DataSocket® Server que simplifica a comunicação através da Internet, gerenciando a programação do protocolo TCP/IP.

As funcionalidades do DataSocket® API são implementadas como controles ActiveX, uma biblioteca em linguagem C do LabWindows/CVI [33] e um conjunto de VIs do LabVIEW®, que podem ser utilizadas nas aplicações. As ações básicas são: abrir, ler, escrever e fechar, similares às operações de chamadas de Entrada/Saída. O mesmo DataSocket® API pode ler dados de diversas localidades:

- Dados em servidores HTTP
- Dados em servidores FTP
- Dados em arquivos locais
- Dados em Servidores de Controle de Processos (OPC)
- Dados em servidores DSTP

O dstp (“DataSocket Transport Protocol”) é um protocolo nativo para as conexões onde um VI deve se comunicar diretamente com o servidor DataSocket®.

O DataSocket® converte automaticamente os dados medidos de uma aplicação em um conjunto de bytes e os envia através da rede. A aplicação cliente converte os dados na forma original, e estas conversões automáticas eliminam muito código que deveria ser escrito, caso fossem utilizadas as bibliotecas de programação de leitura e escrita do protocolo TCP/IP.

A Figura 33 mostra os módulos envolvidos, sendo que o DataSocket® Server é um componente que funciona no servidor, com o qual os programas utilizando o

DataSocket® API podem transmitir os dados através da rede e da Internet, para diversos clientes remotos.

Figura 33 – Utilização do DataSocket® para compartilhamento de dados, através de aplicação em LabVIEW® e a publicação em uma página WEB.

O DataSocket® Server é o programa que recebe e envia os dados para o cliente, sendo que para o acesso ao servidor deve ser fornecido, o endereço IP ou o nome do computador onde o DataSocket® Server está instalado, como mostra a Figura 34.

Figura 34 – Diagrama de um VI com escrita e leitura, utilizando DataSocket®

3.4.2 Características e limitações do Laboratório Virtual com o DataSocket®

O DataSocket® Server pode ser executado na mesma máquina que o programa aplicativo, ou em máquinas separadas. As vantagens de se colocar o DataSocket® Server numa máquina separada do aplicativo são a flexibilidade e a segurança. Uma primeira vantagem reside na segurança, pois se o computador que faz a aquisição apresentar algum problema e sair da rede, o servidor continuará sendo executado, e os clientes ainda terão uma conexão válida, e continuarão a receber os dados, assim que o aplicativo de aquisição volte a funcionar. A segunda vantagem está relacionada com a flexibilidade, pois o servidor separado poderá atender um número maior de clientes, e o aumento de manipulação para a transmissão de dados, não acarretará nenhuma interferência na taxa de utilização da CPU do computador que faz a aquisição. Uma terceira vantagem é que o DataSocket® pode ser colocado fora de uma proteção de “*firewall*”, enquanto o aplicativo permanece sob a proteção segura. Desta maneira pode-se enviar dados por fora de um “*firewall*”, sem a necessidade de uma conexão direta com uma máquina de aquisição de dados.

3.4.3 Velocidade de transferência

O DataSocket® é construído no topo da pilha do protocolo TCP/IP, o que faz com que sua velocidade seja extremamente dependente da largura de banda da rede e do tráfego na rede [xx]. O DataSocket® apresenta uma pequena sobrecarga no

TCP/IP, e alguns testes de limites preliminares obtiveram dados sendo transferidos à velocidades de até 320 KB/s através de máquinas conectadas por redes 10 baseT. A grande limitação da habilidade do DataSocket[®] em transferir dados está na própria limitação da rede utilizada.

3.4.4 Tamanho dos pacotes de dados

O DataSocket[®] foi testado em laboratório, em ambiente com o sistema operacional Windows, com pacote de dados superiores a um Mbyte de tamanho. Apesar de não existir nenhuma restrição técnica que limite o tamanho dos dados transferidos, sua performance tende a diminuir com o aumento do tamanho do pacote de dados.

O DataSocket[®] permite que um total de 1000 clientes de leitura e escrita estejam conectados em um DataSocket[®] Server. Este parâmetro pode ser configurado, além de se definir também quais máquinas podem ser clientes de escrita e leitura [33].

3.4.5 Formato de dados suportados pelo DataSocket[®]

O DataSocket[®] suporta ler os seguintes formatos de dados:

- Texto -txt (assim como os exportados pelo Microsoft Excel)
- Arquivos wav (assim como muitos arquivos de som)
- DSD (DataSocket[®] Data)

O DataSocket[®] suporta escrever os dados nos formatos:

- Texto - txt
- DSD (DataSocket[®] Data)

3.4.6 Criptografia e segurança de dados

O DataSocket[®] não provê diretamente suporte para criptografia dos dados, mas caso seja necessário, a criptografia poderá ser realizada de duas formas. A primeira consiste em implementar um sub VI que o faça, onde os dados são criptografados pelo programa e colocados em um vetor ou matriz, e do outro lado este mesmo esquema de criptografia deverá ser utilizado para decriptografá-lo.

A outra opção é utilizar um programa de terceiros como solução, que automaticamente criptografa e decriptografa os dados ao nível do protocolo TCP/IP.

O DataSocket[®] provê dois métodos para garantir a transferência dos dados:

- DataSocket[®] server manager - que permite que se configure quais máquinas tem a permissão de receber e enviar dados para o DataSocket[®] server.
- DataSocket[®] também permite que seja rodado por fora de um “*firewall*” (aplicativo para segurança), desta maneira o programa, aplicativo, que envia dados está por dentro do “*firewall*” seguro.

O DataSocket[®] suporta ser utilizado sob um “*firewall*” desde que se use um programa chamado SocksCap32 que é disponível pela NEC sem ônus. [33]

3.5 O LabVIEW[®] e o protocolo TCP/IP

O protocolo TCP/IP é utilizado para a comunicação entre equipamentos através de uma rede local simples ou ainda interligando diversas redes, e um exemplo disto é a Internet. O LabVIEW[®] possui blocos que agregam as funcionalidades necessárias para comunicação entre equipamentos utilizando o protocolo TCP/IP, com os quais podem ser construídas aplicações de rede, sem que o programador se preocupe com a complexidade da comunicação. O processo de comunicação envolve abrir uma conexão, ler e/ou escrever dados e fechar a conexão. Os blocos disponíveis são: o **TCP Open Connection** – para abrir a conexão, o **TCP Write** – para escrever dados, o **TCP Read** – para ler os dados e o **TCP Close Connection** – para fechar a conexão.

A Figura 35 mostra um diagrama básico com as funções utilizadas no LabVIEW[®], onde é aberta uma conexão com o endereço IP 127.0.0.1 que é a interface de “*loopback*” do equipamento, e o endereço de uma porta TCP, que no exemplo é a porta (também conhecida como *socket*) 5020.

Figura 35 – Comunicação TCP/IP utilizada no LabVIEW®

Na maioria das comunicações de entrada/saída, o equipamento cliente é que inicia a conexão, buscando acesso a um servidor de arquivos, a um servidor de instrumentos externo ou ainda a um servidor com uma placa DAQ. Com as conexões do protocolo TCP/IP, um computador pode trabalhar como cliente ou ainda como servidor. No diagrama de blocos anterior, a aplicação cliente solicita uma conexão com um servidor remoto com o bloco TCP Open Connection. O servidor, ou um “daemon” escuta a solicitação remota e responde de forma apropriada. Como qualquer cliente pode solicitar uma comunicação com o servidor, deve-se implementar o controle de acesso. O diagrama da Figura 36 mostra como o servidor utiliza o endereço remoto para que o TCP Listen VI determine se um cliente remoto tem permissão de acesso ao servidor.

Figura 36 – Diagrama com detalhamento da VI TCP Listen

As aplicações fazem mais do que ler e/ou escrever um dado. A comunicação é um processo que envolve a utilização de um protocolo. Como exemplo, um cliente envia quatro comandos para o servidor, em formato de oito bits inteiros, sendo: 1 – para aquisição do dado e confirmação, 2 – para enviar dado, 3 – para leitura do estado (“status”) e 4 – para fechar a conexão.

Na Figura 37 o laço “while” envolve todo o restante do VI, que pode trabalhar com múltiplas conexões sequenciais sem a necessidade de reiniciar quando uma conexão é fechada. O VI não pode manipular múltiplas conexões simultâneas. Uma outra estrutura “case” determina se uma conexão válida ocorreu. Se uma conexão válida ocorreu, o VI entra no laço “while” que lê um byte vindo da porta TCP/IP. Este byte recebe comandos de um até quatro, vindo do cliente. Se nenhum comando válido é recebido dentro do período válido para leitura, o caso padrão interno da estrutura “case” envia um valor “TRUE” (Verdadeiro) para o terminal de continuação e assim mantém a conexão ativa.

Figura 37 – Controle de múltiplas conexões com TCP/IP

O diagrama de blocos apresentado na Figura 38 mostra as quatro possibilidades que podem ocorrer na estrutura “case” interna, as quais trabalham com um comando específico que pode ser enviado para o servidor. Cada estrutura envia uma informação através do terminal de continuação, determinando se é para continuar ou não executando o laço. O caso “Quit” sempre irá retornar um valor FALSE (Falso). Após sair do laço, o programa executado no servidor fecha as conexões com o cliente.

Figura 38 – Possibilidades de comandos para a estrutura *case* do VI

A arquitetura cliente/servidor possibilita o desenvolvimento de servidores flexíveis para aplicações complexas utilizando comunicação em rede. As aplicações mais comuns incluem o envio de dados de aquisição para localidades remotas e o controle da aquisição de dados a partir de localidades remotas.

3.6 Considerações finais

Este capítulo descreveu a ferramenta utilizada no desenvolvimento do Laboratório Virtual desta dissertação, mostrando suas características principais e a facilidade de desenvolvimento dos instrumentos virtuais, acelerando os processos de desenvolvimento e manutenção de sistemas que utilizem conceitos de instrumentação, o que a torna bastante interessante para as instituições de ensino, pesquisa e a indústria em geral. A implementação de códigos, chamado VI – instrumento virtual - que representam os equipamentos e o compartilhamento em redes de computadores possibilita a modelagem e o desenvolvimento de projetos para as mais diversas áreas do conhecimento a custos baixos.

4 HISTÓRICO DO PROTOCOLO IP

Este capítulo apresenta conceitos básicos sobre o protocolo IPv6 descrevendo as suas características e funcionalidades, comparando com o protocolo IPv4. Para a implementação de sistemas de rede de computadores utilizando o protocolo IPv6 são apresentados os possíveis cenários de transição.

4.1 Conceitos básicos sobre protocolo IP

O protocolo IP (Internet Protocol) originado nas redes de computadores do Departamento de Defesa norte-americano, no final da década de 70, definido formalmente em 1981, sendo denominada ARPANET – Advanced Research Projects Agency Network, a primeira rede a ter este protocolo implementado. Atualmente o protocolo IP consolidou-se na maior rede de computadores do mundo, responsável pela evolução da Internet, e dos negócios gerados com aplicações distribuídas, cliente – servidor e comércio eletrônico.

O sucesso do protocolo IP, e da família de protocolos TCP/IP (Transfer Control Protocol/Internet Protocol) ocorreu devido a características como simplicidade de arquitetura, orientação a datagrama, qualidade das implementações, modelo cooperativo de interoperabilidade e o suporte a uma grande diversidade de meios de comunicação.

No protocolo TCP/IP versão quatro (IPv4) foram especificadas três classes de endereços para oferecer flexibilidade no endereçamento de redes de várias dimensões, como mostrado na Tabela 2.

Tabela 2- Espaço de endereçamento do protocolo IPv4

Classe	Formato	Nº endereços por rede	Capacidade de endereçamento
A	7 bits de rede, 24 bits de host	16.777.216	2.147.483.648
B	14 bits de rede, 16 bits de host	65.536	1.073.741.824
C	21 bits de rede, 8 bits de host	256	536.870.912
D,E	Multicast experimental	--	--

A existência de classes de endereços fixas é um fator limitante, que leva a uma utilização ineficiente do espaço de endereçamento disponível, tendo como exemplo uma empresa que possua de 300 a 400 equipamentos para serem ligados à Internet – ela irá necessitar de um endereço de classe B já que os da classe C não tem capacidade de endereçamento para atender a necessidade. A empresa utiliza apenas 0,5% da capacidade de endereçamento total da classe B, sendo que os 99,5% restantes são desperdiçados.

Com a constante evolução tecnológica, o poder de processamento dos microcomputadores e a ampliação dos negócios sendo realizados através da Internet, o protocolo IPv4 conseguiu suportar as mudanças, se tornando o padrão de comunicação, apesar de não ter sido originalmente projetado para oferecer suporte a uma rede de escala universal, ou ainda, com a possibilidade da utilização de aplicações multimídia [34]. Uma atualização de protocolo de comunicação na Internet começa a surgir.

Em 1991, membros do **IETF** (*Internet Engineering Task Force*) [IETF] concluíram que o crescimento exponencial da rede teria como consequência, a escassez dos endereços IP, e ainda assim se as tabelas de roteamento não tivessem sua capacidade esgotada nos hardwares de roteamento da época.

A adoção de políticas de atribuição de endereços utilizando o **CIDR** (*Classless Inter-Domain Routing*), visava alocar blocos de endereços da Classe C contíguos para grandes regiões, e estas dividiriam seus blocos em blocos menores, mas ainda contíguos, até que todas as redes tivessem seus endereços. Com a utilização de máscara de rede, os roteadores usariam uma máscara para endereçar todo um bloco de endereços e assim diminuir a tabela de roteamento [26]. Esta solução apenas adiou a data da escassez do espaço de endereçamento, e um novo protocolo precisava ser desenvolvido em substituição ao IPv4. Uma proposta foi adotar o CLNP [26], que tem um espaço de 160 bits para endereçamento, porém, o fato de não suportar serviços multimídia como desejado, e por ser uma solução OSI, não foi aceita.

Em 1993, o **IESG** (*Internet Engineering Steering Group*) [IESG] nomeou um grupo de trabalho com a responsabilidade de especificar uma nova versão do protocolo IP, o **IPngWG** (*IP Next Generation Working Group*) [IPNG]. Foram

selecionados alguns protocolos para a camada de rede da arquitetura TCP/IP. O protocolo vencedor foi o **SIPP** (*Simple Internet Protocol Plus*) [R&D 95], por ser muito parecido com o IPv4, e ter um plano de transição melhor. Uma combinação de aspectos positivos dos três protocolos que foram analisados foi feita, e com isso a recomendação para a versão seis do protocolo IP, em novembro de 1994.

O protocolo IPng foi projetado como uma evolução do protocolo IPv4, mantendo as funcionalidades mais importantes, e as que não possuíam bom desempenho ou que eram pouco utilizadas foram retiradas ou tornadas opcionais.

4.2 O protocolo IPv6 – Conceitos

Para se estudar e implementar algo novo, como a arquitetura do protocolo IPv6, é necessário verificar as principais características do protocolo IP implementado, atualmente na versão quatro, suas limitações face às novas necessidades de serviços e analisar a documentação concernente ao IPv6, através das RFCs – Request for Comment: RFC 2373 – IP Version seis Addressing Architecture [35], a RFC 2460 – Internet Protocol, Version seis (IPv6) Specification [36], a RFC 1886 – DNS Extensions to Support IPv6 [37], a RFC 2471 – IPv6 Testing Address Allocation [38], a RFC 2893 – Transition Mechanisms for IPv6 Hosts and Routers [39], a RFC 2461 – Neighbor Discovery for IP Version seis (IPv6) [40], a RFC 2185 – Routing Aspects of IPv6 Transition [41], que formam a base da arquitetura do protocolo.

Os objetivos que devem ser alcançados com a nova versão do protocolo IP são [26]:

- **Suporte a bilhões de hosts** – o tamanho de endereços de 32 bits do IPv4 passa para 128 bits na versão IPv6, o que significa um espaço muito maior de nós finais, hierarquização melhor do espaço de endereçamento, essencial a escalabilidade do roteamento e uma melhor facilidade em termos de autoconfiguração, permitindo utilização de endereços IEEE-802 embebidos de endereços IPv6 [42]. A utilização de um campo que define o âmbito do alcance do datagrama, aumenta a escalabilidade dos endereços multicast.

- **Simplificação do cabeçalho do protocolo, diminuindo o tempo de processamento:** alguns campos do cabeçalho do protocolo IPv4 foram retirados ou passaram a ser opcionais, simplificando o tratamento de um pacote comum. Embora os endereços IPv6 sejam quatro vezes maiores em tamanho do que os endereços IPv4, o cabeçalho é apenas duas vezes maior, isto é, utiliza 40 bytes [42].
- **Protocolo passível de expansão, através do uso de cabeçalhos de extensão:** As opções são codificadas em cabeçalhos separados no IPv6, localizadas entre o cabeçalho IPv6 e o cabeçalho da camada de transporte [42]. Sendo a maioria das opções processadas por nós finais, esta codificação permite que a utilização de opções e extensões ao protocolo, não interfira com a capacidade de encaminhamento de pacotes nos roteadores. Este suporte possibilita a incorporação de opções futuras, permitindo assim maior flexibilidade.
- **Redução da tabela de roteamento:** na análise dos cabeçalhos, realizada pelos roteadores e hosts. [43]
- **Garantia de maior segurança:** utilizando autenticação e privacidade em relação à versão atual [44] [45] [46].
- **Criação de um campo que suporte mecanismos de controle de qualidade de serviço:** gerando maior sensibilidade ao tipo de serviço, como, por exemplo, serviços de tempo real [47].
- **Permissão de *multicasting*:** através da especificação de escopos de sessões *multicasting* [47].
- **Melhorias no roteamento,** inclusive no tratamento aos equipamentos (hosts) móveis.
- **Permissão de máquinas *wireless* trocarem fisicamente de lugar:** sem a necessidade de alteração na configuração de seus endereços IP [48].
- **Habilitação das máquinas se autoconfigurarem** (número IP, servidor de nome...) ao serem ligadas na rede, como em uma operação "plug and play" [49] [50].

- Um novo tipo de endereço chamado *anycast*: que identifica um conjunto de nós, onde um pacote enviado para um endereço *anycast* será entregue a um destes nós [51].
- **Coexistência das duas versões do protocolo:** não se determinando uma data específica para que todas as máquinas no mundo troquem seus softwares [52].

4.3 Cabeçalhos dos protocolos IPv4 e IPv6

Vamos analisar as características dos campos dos cabeçalhos do protocolo IPv4 e IPv6, procurando observar as vantagens que foram implementadas e as novas funcionalidades.

Na Figura 39 encontramos o formato do cabeçalho do protocolo IPv4.

Figura 39 - Cabeçalho do protocolo IPv4

Versão (Version) – 4 bits, controla a versão do protocolo a que o datagrama pertence.

Comprimento do cabeçalho (IHL) – 4 bits, como o tamanho do cabeçalho no IPv4 não é constante, este campo com valor mínimo cinco, significando a ausência de opções, e valor máximo 15, limitando o cabeçalho a 60 bytes e o campo de opções a 40 bytes. Para algumas opções, como a que registra a rota percorrida por um pacote, este tamanho não é satisfatório, tornando a opção inútil [27].

Tipo de serviço (Type of Service) – 8 bits, possibilita que o host informe à sub-rede o tipo de rede que necessita, com vários níveis de confiabilidade e velocidade.

Comprimento total (Total length) – 16 bits, inclui tudo que existe no datagrama, cabeçalho e dados, sendo o tamanho máximo de 65535 bytes.

Identificação (Identification) – 16 bits, permite ao host de destino determinar a qual datagrama pertence um fragmento recém-chegado; sendo que todos os fragmentos de um mesmo datagrama devem ter o mesmo identificador.

FLAGS – 3 bits, sendo o primeiro não utilizado, o segundo indica para os roteadores não fragmentarem o datagrama, pois a máquina de destino não consegue juntar os pedaços, e o terceiro bit indica que existem mais fragmentos. Em todos os fragmentos, exceção feita ao último, existe este conjunto de bits, necessários para sinalizar para a máquina destino que todos os fragmentos já chegaram [27].

Deslocamento de fragmentação (Fragment Offset) – 13 bits, indica a que ponto do datagrama atual o fragmento pertence. Os fragmentos de um datagrama, exceto o último, devem ser múltiplos de oito bytes, que é a unidade de fragmento.

Tempo de vida (Time to live) – 8 bits, é utilizado como um contador para limitar o tempo de vida do pacote, sendo seu valor máximo de 255 segundos, devendo ser incrementado em cada hop, e decrementado diversas vezes, quando é enfileirado por um longo tempo em um roteador. Ele, na prática, conta os hops, e quando seu valor chega a zero, o pacote é descartado e um pacote de advertência deve ser enviado ao host de origem.

Protocolo (Protocol) – 8 bits, utilizado para indicar quando a camada de rede já tiver montado um pacote completo, e informar o processo de transporte a ser aplicado ao datagrama, podendo ser TCP ou UDP, entre outros.

Verificação de cabeçalho (Header checksum) – 16 bits, utilizado para a verificação da integridade do cabeçalho.

Endereço origem (Source Address) e Endereço de destino (Destination Address) – 32 bits, indicam o número de rede e o número do host, de origem e destino do datagrama, respectivamente, e a quantidade de bits, que especifica, a rede e o host, dependem da classe a que o endereço IP pertença.

Opções (Options) – 32 bits, utilizado para permitir que novas funcionalidades sejam colocadas, sem a necessidade de inserção de bits de cabeçalho, sendo preenchido por um múltiplo de quatro bytes [27].

Na Tabela 3 estão as opções definidas para o campo Opções, que incluem segurança, caminho completo desde a origem até o destino, como uma seqüência de endereços IP, lista de roteadores que o pacote deve passar, registro do endereço e do *timestamp* do roteador.

Tabela 3 - Características do campo Opções do protocolo IPv4

Opção	Descrição
Security	Define o nível de segurança do datagrama
Strict source routing	Fornecer o caminho completo a ser seguido
Loose source routing	Mostra lista de roteadores pelos quais o pacote deve passar
Record route	Indica que cada roteador deve anexar seu endereço IP ao campo de opção
Time Stamp	Indica que cada roteador deve anexar seu Time Stamp

O protocolo IPv6 altera o formato do datagrama IP. A Figura 40 mostra que um datagrama IPv6 tem um cabeçalho base fixo, seguido de nenhum ou mais cabeçalhos extras, e depois os dados [53].

Figura 40 – Formato do datagrama IPv6

O protocolo IPv6 possui um cabeçalho relativamente simples [54], com menos informações que o do IPv4, sendo que, alguns campos do cabeçalho do IPv4 e

as opções foram substituídos por cabeçalhos de extensão, utilizados quando necessários.

Na Figura 41 temos os campos do cabeçalho do protocolo IPv6.

Figura 41 - Cabeçalho do protocolo IPv6

Versão – 4 bits, utiliza o valor seis para a versão IPv6 e o valor quatro para a versão IPv4. Será utilizado durante o período de transição para que os roteadores examinem este campo e identifiquem o tipo de pacote [27].

Classe de tráfego (Priority) – 4 bits, utilizado para determinar a prioridade do datagrama em relação a outros datagramas da mesma origem. Todos os pacotes, de determinado fluxo devem ter a mesma prioridade, logo estes são dois campos usados em conjunto. Espera-se que esse campo identifique e priorize aplicações iterativas, como por exemplo, uma sessão remota.

O uso efetivo ocorre quando o pacote enfrenta um tráfego congestionado. Os valores de zero a sete nesse campo lidam com transmissões (geralmente TCP) que podem ser retardadas no caso de um congestionamento. Os valores de 8 a 15 se referem a aplicações cujo tráfego é constante, e um atraso implicaria em perda de informação, como por exemplo, vídeo e áudio [27].

Rótulo de fluxo (Flow Label) - 24 bits, que podem ser utilizados para identificar um tipo de fluxo de dados [34]. Uma utilização prevista será a de atribuir um tratamento diferenciado ao pacote, em casos de tráfego e em congestionamentos. Alguns serviços, ao ter este campo adequadamente configurado, podem ser tratados com Qualidade de Serviço (QoS) diferente.

Tamanho da carga (Payload Length) – 16 bits, identifica o número de bytes que se seguem após o cabeçalho de 40 bytes do IPv6, ou seja, o resto do pacote que segue ao cabeçalho do IPv6.

Próximo cabeçalho (Next Header) – 8 bits, identifica o próximo cabeçalho, se existir como cabeçalho de extensão do IPv6, ou caso não exista. Indica o protocolo acima da camada de rede. Esta é uma estratégia para simplificar o cabeçalho do protocolo IPv6.

Limite de hop (Hop Limit) – 8 bits, utilizado para indicar o número máximo de hops através dos quais o pacote pode trafegar, sendo decrementado a cada hop pelo qual o pacote passa. Ao chegar em zero, o pacote é descartado.

Endereço de origem (Source Address), Endereço de destino (Destination Address) – 128 bits, identificam o endereço origem e destino, respectivamente, do pacote, sendo que os endereços que começam com 80 zeros são reservados para endereços IPv4.

Com o avanço das telecomunicações e as novas aplicações que vão surgindo, o protocolo deve proporcionar tratamento diferenciado para os pacotes que trafegam pelas redes. Na estrutura do cabeçalho do protocolo IPv6, os campos de *Rótulo de fluxo* e *Classe de tráfego* são utilizados para identificar os pacotes que necessitam de "tratamento diferenciado", por exemplo, os pacotes gerados de aplicações multimídia ou de tempo real.

4.3.1 Cabeçalhos de extensão do protocolo IPv6

O protocolo IPv6 possui ainda a opção de inserir cabeçalhos de extensão, quando necessários, para oferecer alguma funcionalidade. Atualmente estão definidos seis cabeçalhos de extensão, com informações para os roteadores, definição de rota parcial ou integral a ser seguida pelo pacote, gerenciamento de fragmentação, verificação da identidade do transmissor, informações sobre o conteúdo criptografado e informações adicionais para o destino.

Um pacote sendo transmitido poderá ter mais cabeçalhos dentro dele. Num cabeçalho IPv6, no qual onde o campo de próximo cabeçalho indica que o pacote deve ser roteado, tem como cabeçalho de próximo nível, colocado depois do cabeçalho IPv6, o cabeçalho de roteamento; e o cabeçalho da próxima camada

poderá ser o TCP. Ou ainda, o cabeçalho de roteamento pode indicar o fragmento no campo de próximo cabeçalho e assim por diante.

Estes cabeçalhos são inseridos entre o cabeçalho IPv6 e os cabeçalhos da camada 4, mostrado na Figura 42.

Figura 42 – Exemplo de inserção de cabeçalho de extensão

Os cabeçalhos de extensão para o protocolo IPv6 são definidos através da RFC 2460, da seguinte forma:

- Opções de hop a hop;
- Roteamento (tipo zero);
- Fragmento;
- Opções de destino;
- Autenticação;
- Carga de segurança de encapsulamento.

Um nó receptor não deverá verificar o pacote todo, procurando por um cabeçalho de extensão em particular, e processá-lo. Quando existir um cabeçalho de extensão, com a opção hop a hop, este será examinado por todos os nós ao longo do caminho de entrega do pacote, e se este cabeçalho estiver presente, virá logo depois do cabeçalho IPv6. O valor zero, no campo de próximo cabeçalho, indica a sua presença em um pacote.

A Figura 43 mostra a ordem em que os cabeçalhos de extensão são organizados num pacote. O cabeçalho de opções de destino-1 é processado pelo primeiro nó destino e mais destinos que estejam listados no cabeçalho de roteamento. O cabeçalho de opções de destino-2 é processado apenas pelo nó final.

Figura 43 – Sequência dos cabeçalhos de extensão no pacote IPv6

O cabeçalho de extensão é um múltiplo de um inteiro com oito octetos de extensão. As opções dos cabeçalhos de hop a hop e de destino são codificadas por TLVs (tipo-comprimento-valores), com o formato incluindo um campo de tipo de

opção (oito bits), um campo com o comprimento dos dados da opção (oito bits) e um campo de dados (com tamanho variável).

O identificador de tipo de opção é codificado de modo que seus dois bits de ordem mais alta determinam a ação a ser realizada pelo nó, processando o pacote IPv6, não reconhecer o tipo de opção.

O terceiro bit de ordem mais alta especifica se os dados da opção mudam ou não o caminho. A Figura 44 mostra o TLV do cabeçalho de extensão.

Figura 44 – TLV de um cabeçalho de extensão

4.3.2 Cabeçalho de opção hop a hop

O cabeçalho hop a hop, transporta opções, que são verificadas por todos os nós no caminho percorrido pelo pacote, sendo o primeiro cabeçalho de extensão, depois do cabeçalho IPv6, e é identificado pelo valor zero, no campo de próximo cabeçalho; o próximo cabeçalho, dentro do cabeçalho de extensão, é um campo de oito bits, que identifica o tipo de cabeçalho, que começa logo após o cabeçalho de opção hop a hop. O próximo campo identifica o tamanho da opção hop a hop em oito octetos, tendo um tamanho variável com opções codificadas por TLV.

A Figura 45 mostra o cabeçalho de opções hop a hop.

Figura 45 – Cabeçalho de opções hop a hop

Cada roteador verifica o cabeçalho hop a hop, e isto possibilita então, que sejam transmitidas informações de gerenciamento ou comandos de depuração para os roteadores.

4.3.3 Cabeçalho de opções de destino

O cabeçalho de opções de destino é utilizado para transportar informações que devem ser analisadas pelo nó de destino, sendo que o valor 60 (sessenta), identifica este tipo de cabeçalho.

As informações de destino, opcionais, podem ser codificadas em um pacote IPv6 ou por meio de uma opção no cabeçalho de opções de destino, ou ainda, através de um cabeçalho de extensão separado, que é um cabeçalho de fragmento e um cabeçalho de autenticação. Os dois bits de mais alta ordem do campo tipo de opção indicam a ação a ser tomada.

A Figura 46 mostra o formato do cabeçalho de destino e o significado dos bits de mais alta ordem.

Figura 46 – Formato do cabeçalho de opção de destino

4.3.4 Cabeçalho de roteamento

O cabeçalho de roteamento é utilizado por um nó de origem IPv6, para identificar a lista de um ou mais hosts intermediários a serem visitados no caminho de entrega do pacote em direção ao destinatário. O valor 43 (quarenta e três) no campo próximo cabeçalho indica ser o cabeçalho de roteamento.

O cabeçalho de roteamento tipo zero (zero) não deverá conter um endereço de multicast no campo de endereço. O próximo cabeçalho, depois do tipo de rota, é o campo de segmentos restantes, indicando o número de rotas que ainda faltam, isto é, o número de nós a serem visitados antes de alcançar o destino.

O formato do cabeçalho de roteamento, com seus campos é mostrado na Figura 47.

Figura 47 – Formato do cabeçalho de roteamento tipo 0

4.3.5 Cabeçalho de fragmentação

O cabeçalho de fragmentação, para o protocolo IPv6, é utilizado pela origem para enviar um pacote maior do que a unidade máxima de transmissão (MTU). No protocolo IPv4 a fragmentação é realizada no roteador. O IPv6, conforme descrito na RFC 2460 e na RFC 1981, exige um MTU de 1280 bytes ou mais em cada link. Para o caso de um link não conseguir transmitir um pacote de 1280 bytes, num só local, a fragmentação específica deverá ser realizada pelos protocolos básicos, e não pelo IPv6. A recomendação é que caso exista a possibilidade, o MTU de um link deve ser configurado para 1500 bytes, acomodando um possível encapsulamento. O cabeçalho de fragmento é identificado pelo valor 44 (quarenta e quatro) no campo próximo cabeçalho.

Um pacote pode conter duas partes, uma não fragmentável e a outra parte fragmentável, que pode ainda, ser dividida em várias partes. O nó de origem fragmenta o pacote, se este for maior que a MTU aceita pelo nó destino, sendo que cada pacote fragmentado é identificado com um identificador exclusivo, e o nó de destino irá recuperar o pacote unindo os fragmentos de forma adequada, de acordo com os identificadores de fragmento, como mostrado na Figura 48.

Figura 48 – Fragmentação de pacote por nó de origem do IPv6

4.3.6 Cabeçalho de Autenticação

O cabeçalho de autenticação (AH) é o meio de fornecer integridade e autenticação forte e sem conexão para os datagramas IP. Utilizado com o cabeçalho ESP (carga de segurança do encapsulamento IP), pode proporcionar serviços de segurança misturados no IPv4 e no IPv6. O cabeçalho AH mostrado na Figura 49, é identificado pelo campo de protocolo (IPv4) e o valor 51 (cinquenta e um) no campo IM (próximo cabeçalho), antes dos cabeçalhos de protocolo (IPv4, IPv6 ou extensão).

Próximo cabeçalho (8 bits)	Tamanho de carga útil (8 bits)	RESERVADO (16 bits)
Índice do parâmetro de segurança (SPI) (32 bits)		
Número Sequencial (32 bits)		
Dados de autenticação (tamanho variável)		

Figura 49 – Formato do cabeçalho de autenticação (AH)

Onde:

Próximo cabeçalho – identifica o tipo de carga útil após o cabeçalho de autenticação (AH), sendo um conjunto de números definidos pela RFC 1700.

Tamanho de carga útil – identifica o tamanho do cabeçalho de autenticação na palavra de 32 bits, sendo que um valor zero neste campo indica que ele é usado para fins de depuração. Para o protocolo IPv4, o valor nesse campo é um e para o IPv6, o valor é dois.

Reservado – é um campo reservado para uso futuro, e definido como zero.

Índice do parâmetro de segurança (SPI) – é um valor arbitrário, que identifica a associação de segurança dos datagramas IP, em conjunto com o protocolo DA e segurança IP (cabeçalho de autenticação). O valor zero, reservado para uso local.

Número seqüencial – inclui um contador seqüencial, que sempre aumenta, e está presente no pacote, e é enviado pelo nó de origem.

Dados de autenticação – este campo inclui o valor de verificação de integridade (ICV) para o pacote, e é calculado baseando-se nos campos do cabeçalho (que são absolutos em trânsito e, portanto, previsíveis na chegada); no cabeçalho de autenticação (AH) e bytes de enchimento explícitos e os dados de protocolo de nível superior, também absolutos em trânsito.

Para o IPv4, o cabeçalho de autenticação (AH) é colocado logo após o cabeçalho IPv4 e o cabeçalho do nível superior, por exemplo, o cabeçalho TCP. Para o IPv6, o cabeçalho de autenticação (AH) é considerado como carga útil de ponta a ponta, e é colocado após os cabeçalhos de hop a hop, roteamento e fragmentação.

4.3.7 Cabeçalho de carga de segurança de encapsulamento

O cabeçalho ESP (Encrypted Security Payload) possibilita a transmissão de dados criptografados pelo IPv6, sendo utilizado para autenticar a origem dos dados, a confidencialidade, a integridade, e uma forma de serviço de integridade de seqüência, denominada anti-replay.

O valor 50 (cinquenta) no campo de protocolo para IPv4, e o campo de próximo cabeçalho, para o IPv6, identifica o ESP como seqüencial e esse cabeçalho.

A Figura 50 mostra o formato do cabeçalho ESP (de criptografia IPv6) e seus campos.

Figura 50 – Formato do cabeçalho de segurança de encapsulamento

Onde:

Índice do Parâmetro de Segurança (SPI) – esse campo, em combinação com o DA e um protocolo de segurança como ESP, identifica a associação de segurança do datagrama, possuindo valor que varia de 1 a 255, reservado pela IANA, e é um campo obrigatório, usado pelo sistema de destino no estabelecimento da associação de segurança. O valor zero é reservado para o uso local, não devendo ser enviado pela linha.

Número Sequencial - esse campo inclui um valor de contador ou número de seqüência, crescente, e obrigatório. É enviado pelo emissor e o processamento desse campo, depende do receptor, que pode decidir atuar ou não sobre ele.

Dados da carga útil - esse é um campo que contém dados descritos pelo próximo cabeçalho; é um campo obrigatório e pode transportar um vetor de inicialização (IV) usado para criptografia da carga útil.

Enchimento - utilizado para a criptografia e para justificar a presença do campo de enchimento, dependente das seguintes condições:

- Se um algoritmo de criptografia for implementado, e necessitar que o texto puro tenha um número múltiplo de bytes, o enchimento preenche o texto puro até o tamanho exigido pelo algoritmo.

- Se a criptografia plana não for utilizada, o enchimento então garantirá que o texto cifrado termine num limite de quatro bytes.

- O enchimento, pode ainda ser utilizado para esconder o tamanho da carga útil, oferecendo suporte a confidencialidade parcial do fluxo de tráfego.

Tamanho do enchimento - Esse campo indica o número de bytes no campo de enchimento anterior a ele. O intervalo válido é de 0-255, onde o valor zero indica que nenhum byte de enchimento estará presente.

Próximo cabeçalho - esse campo indica o tipo de dado, contido no campo de dados da carga útil. Os dados da carga útil podem ser um cabeçalho de extensão ou um identificador de protocolo do nível superior.

Dados de autenticação - esse campo inclui um ICV, calculado por meio do pacote ESP menos os dados de autenticação. É um campo opcional, utilizado apenas se o serviço de autenticação for selecionado para a associação da segurança.

4.4 Arquitetura de Endereçamento do protocolo IPv6

O espaço de endereçamento de 32 bits no protocolo IPv4 passa a ser de 128 bits, na versão IPv6, logo enquanto IPv4 suportava quatro bilhões de nodos, o IPv6 pode endereçar 3.4×10^{38} nodos [55].

O protocolo IPv6 acaba com a utilização de classes de endereços, permitindo um método mais simples de autoconfiguração, sendo baseado em prefixos como o IPv4, mas alterando a função desses prefixos, que não mais identificam as diferentes classes de endereços, e sim os diferentes usos dos endereços. Os endereços que começam com 80 zeros são reservados para endereçamento IPv4. Duas variantes são suportadas, identificadas nos 16 bits seguintes, que permitirão que o protocolo IPv6 trafegue numa estrutura ainda baseada em IPv4 [26].

Dois conjuntos de endereços são reservados para encapsulamento de protocolos que não sejam o IP, como OSI NSAP e Novell IPX.

A autoconfiguração dos endereços permitirá uma operação *plug-and-play* das máquinas na Internet, e quando a máquina for ligada, deverá associar

automaticamente um endereço IP à sua interface de rede. Com o protocolo IPv4, a associação de endereços IP era feita manualmente, bem como a associação com o servidor de nomes e o “*default gateway*”. Com o protocolo IPv6, a característica de autoconfiguração permite aos equipamentos mudar de sub-rede mais facilmente, o que é muito interessante para as máquinas com tecnologia Wireless.

Existem duas formas de um *host* se autoconfigurar com o protocolo IPv6 [55]:

- **Modo *stateful*** - onde há um servidor de configuração, com o qual o *host* se comunica [56].
- **Modo *stateless*** - onde o *host* constrói seu endereço IP a partir do seu endereço de interface de rede, que é único [49] [50].

Para identificação de empresas com funções de provedores de acesso é utilizado o prefixo *Provider-Based*, disponibilizando a elas, uma grande parte do espaço de endereçamento, para que dividam hierarquicamente, como feito no CIDR [54].

4.5 Notação de Endereços IPv6

Os endereços IPv6 ocupam um espaço de 16 bytes (128 bits), sendo escritos em oito grupos de quatro dígitos hexadecimais cada, separados por dois pontos (:).

**6000:0000:0000:0000:0345:6567:96AD:BEAD
46DC:4231:3211:BC22:0012:3467:FA45:0384**

Para facilitar a escrita e a leitura de endereços IPv6, visto que possuem muitos bytes em 0, foi convencionado:

- Os zeros podem ser omitidos no início do grupo. Desta forma, 0345 pode ser escrito como 345.
- Um ou mais grupos com quatro bytes em 0 podem ser omitidos, sendo substituídos por um par de dois pontos (::).

Exemplo:

O endereço IPv6: **6000:0000:0000:0345:6567:96AD:BEAD:F340**

Pode ser escrito de forma abreviada como:

6000::345:6567:96AD:BEAD:F340

- Os endereços IPv4 podem ser escritos colocando-se um par de dois pontos seguido da notação do endereço no formato octal.

::143.47.1.25

Na Figura 51 temos o formato do prefixo Provider-Based Unicast, e a identificação de cada campo de bits.

010 (3 bits)	REGISTRY ID (N bits)	PROVIDER ID (M bits)	SUBSCRIBER ID (x bits)	SUBNET ID (y bits)	INTERFACE ID (x-y bits)
-----------------	-------------------------	-------------------------	---------------------------	-----------------------	----------------------------

Figura 51- Formato do Prefixo Provider – Based Unicast

A Tabela 4 mostra os prefixos de endereços IPv6 e a sua utilização.

Tabela 4- Prefixos de endereços utilizados no IPv6

Prefixo	Alocação de endereço	Espaço de endereços
00000000	Reservado	1/256
00000001	Não atribuído	1/256
0000001	Alocação NSAP	1/128
0000010	Alocação IPX	1/128
0000011	Não atribuído	1/128
00001	Não atribuído	1/32
0001	Não atribuído	1/16
001	Unicast global	1/8
010	Não atribuído	1/8
011	Não atribuído	1/8
100	Não atribuído	1/8
101	Não atribuído	1/8
110	Não atribuído	1/8
1110	Não atribuído	1/16
11110	Não atribuído	1/32
111110	Não atribuído	1/64
1111110	Não atribuído	1/128
111111100	Não atribuído	1/512
1111111010	Endereço local do link	1/1024
1111111011	Endereço local do site	1/1024
11111111	Endereço Multicast	1/256

Onde:

- Os primeiros três bits (010) do endereço identificam o prefixo, reservado para provedores.
- **REGISTRY ID** – os n bits identificam o registro do provedor na Internet, que pode alocar partes para seus clientes, os assinantes.
- **SUBSCRIBER ID** – os x bits identificam os assinantes naquele provedor.
- **SUBNET ID** – os y bits identificam uma sub-rede, uma ligação física específica.
- **INTERFACE ID** – os $(x-y)$ bits identificam a interface ligada na rede (*host*).

A arquitetura de endereçamento do IPv6, definida pela RFC 2373, introduz três tipos de endereços aplicados a uma rede IPv6, ou seja: endereço unicast, endereço anycast e endereço multicast.

4.6 Hierarquia dos endereços IPv6

O protocolo IPv4 era incapaz de tratar a explosão da tabela de roteamento, para solucionar este problema, o protocolo IPv6 tem um esquema de endereçamento organizado em uma hierarquia, na qual o nível superior inicia com os bits mais significativos do endereço, indo até o bit menos significativo.

O endereço IPv6, composto de 128 bits, tem seus bits, a partir dos primeiros sessenta e quatro bits, definindo o prefixo de formato (FP), os TLA ID (alocados para identificação de agregação do nível de topo), o RES (reservado para futuras expansões), o NLA ID (representando a alocação para grandes provedores), o SLA ID (utilizado para grandes empresas, endereçando as suas sub-redes) e o ID (utilizado para identificar a interface de rede).

Na Figura 52 temos o formato utilizado para a hierarquia de endereços do protocolo IPv6.

FP (3 bits)	TLA ID (13 bits)	RES (8 bits)	NLA ID (24 bits)	SLA ID (16 bits)	INTERFACE ID (64 bits)
----------------	---------------------	-----------------	---------------------	---------------------	---------------------------

Figura 52 - Formato do Aggregatable Global Unicast Address

Onde:

FP – 3 bits, prefixo de formato igual a 001, indica o tipo de endereço que vem a seguir, ou seja, unicast, anycast ou multicast.

TLA ID – 13 bits, é o identificador TLA (Top-Level Aggregation Identifier), que permite alocar 8192 TLAs ID, ou seja, são os pontos de trânsito público (centrais) onde os provedores de telecomunicação estabelecem as conexões ponto a ponto.

RES – 8 bits, é um campo reservado, podendo ser utilizado para expansão do campo TLA ID ou mesmo do campo NLA ID.

NLA ID – 24 bits, é o identificador NLA (Next-Level Aggregation Identifier), que permite alocar 16 milhões de NLAs ID, e representa grandes provedores e redes corporativas globais.

SLA ID – 16 bits, é o identificador SLA (Site-Level Aggregation Identifier), que aloca até 65.535 sub-redes individuais por site.

INTERFACE ID – 64 bits, é o identificador da interface, sendo que seu tamanho foi escolhido para dar suporte a identificadores de interface baseados no formato EUI-64.

Os TLAs são os pontos de trânsito ou provedores de topologia de trânsito, não estando vinculados à inscrição de endereço, mas estabelece responsabilidades por recursos importantes da Internet. A especificação RFC 2450 indica que a implementação de endereços TLA será realizada em dois momentos. Num primeiro momento, um sub-TLA será alocado, e quando o destinatário atribuir os NLAs IDs, o TLA ID será então atribuído para o provedor de topologia de trânsito.

A Figura 53 nos mostra de forma gráfica a técnica usada para atribuição de endereços IPv6.

Figura 53 – Técnica para atribuir endereços IPv6, de forma hierárquica

4.6.1 Endereços Unicast

Este tipo de endereço é assinalado para uma única interface, também disponível no IPv4, sendo que um pacote destinado a um endereço unicast é enviado diretamente para a interface associada ao endereço. Os tipos de endereços unicast foram definidos como: [57]

- **Endereço unicast global (Aggregatable Global Unicast Addresses):** é este endereço unicast que será globalmente utilizado na Internet [58]

Na Figura 54 temos um endereço unicast recebido por um nó final, que o interpreta como seu endereço e por um roteador, que interpreta como uma parte do endereçamento da sub-rede e outra parte como endereço da interface nesta rede.

Figura 54 – Estrutura de endereço unicast recebido por nó e por um roteador

- **Endereço do próprio Nó (Loopback Address):** utilizado quando um nó envia um datagrama para si mesmo, não podendo ser associado a nenhuma interface física, nem de origem e nem de destino. Definido como sendo **0:0:0:0:0:0:1**
- **Endereço não especificado (Unspecified Address):** indica a ausência de endereço, podendo ser utilizado apenas em estações ainda não iniciadas. Não deve ser utilizado em nenhum nó, ou em endereço de destino ou em cabeçalho de roteamento de pacote IPv6. Definido como sendo **0:0:0:0:0:0:0**
- **Endereço IPv4 embutido (Embedded IPv4 Addresses):** é um endereço IPv6 com um endereço IPv4 embutido, denominado IPv4-compatible IPv6 Address. É constituído de um prefixo nulo (96 zeros) e um endereço IPv4 (32 bits), por exemplo, **::173.18.26.37** onde os **::** indicam o prefixo com 96 zeros e o **173.18.26.37** indica um endereço IPv4, mostrado na Figura 55. Será muito útil no período de transição para que os hosts e roteadores possam “tunelar” pacotes IPv6 em estruturas de roteamento IPv4 [59].

Figura 55 – Endereço IPv6 mapeado para IPv4

- **Hosts sem suporte a IPv6:** definido o endereço **IPv4-mapped IPv6 Address** com o valor, por exemplo **::FFFF:173.18.26.37**, utilizado para dar suporte aos nós que não entendem IPv6.
- **Endereço IPX (IPX Address):** com 121 bits, identificado pelo prefixo **0000010**, para permitir mecanismos de mapeamento de endereços IPX utilizados em redes Netware em IPv6.
- **Endereços NSAP (NSAP Addresses):** com 121 bits, identificado pelo prefixo **0000001**, utilizado para suporte de endereçamento OSI NSAP (Network Service Access Point) em redes IPv6.
- **Endereços de uso local (Local-Use IPv6 Address):** utilizados para Link-local e Site-local [60].

Os endereços de uso local (*link* e *site*) são reservados para uso local, para que as máquinas se autoconfigurem ao serem conectadas, sendo que os roteadores não devem repassar pacotes que tenham endereços origem ou destino com estas características.

- **Endereço do Link-local** - reservado para uso local num único enlace para funções de autoconfiguração, descoberta da vizinhança, ou ainda quando não existe roteador.

O formato do endereço de Link-local é mostrado na Figura 56.

1111111010 (10 bits)	00000.....00000 (58 bits)	ID da Interface (64 bits)
-------------------------	------------------------------	------------------------------

Figura 56– Formato do endereço do link local

- **Site-local** – reservado para uso interno em uma empresa que não se conectará na Internet, e não tem a necessidade de um prefixo global, tendo seu formato mostrado na Figura 57.

1111111011 (10 bits)	00000.....00000 (38 bits)	ID da sub-rede (16 bits)	ID da Interface (64 bits)
-------------------------	------------------------------	-----------------------------	------------------------------

Figura 57- Formato do endereço unicast do Site local

Para os tipos de prefixo link local e site local, o *INTERFACE ID*, deve ser único por interface, não podendo se repetir. O campo *SUBNET* identifica uma sub-rede dentro desse domínio.

O uso desse tipo de endereçamento é muito útil para a construção de redes internas, onde podemos construir uma rede baseada na arquitetura TCP/IP sem se ter endereços alocados. A vantagem desse tipo de endereçamento é que enquanto uma rede interna usando o IPv4, ao se conectar na Internet, deve trocar todos os endereços IP das máquinas conectadas a ela, com esse esquema ela pode usar seu endereço de rede e interface em conjunto com um prefixo global alocado a ela, com o prefixo *Provider-Based*.

4.6.2 Endereço Anycast

Um endereço IPv6 *anycast* é um endereço atribuído a mais de uma interface, tipicamente pertencendo a nodos diferentes, alocados a partir de endereços unicast, sendo que um pacote enviado a esse endereço será entregue à interface mais próxima, de acordo com os protocolos de roteamento [54] [51].

Esse tipo de endereçamento pode ser usado por um nó para determinar a rota pela qual ele deseja que seus pacotes trafeguem, com base no nó destinatário. Assim ele poderia selecionar por quais provedores seus pacotes podem passar sem se preocupar com segurança. Essa capacidade pode ser implementada através da configuração de endereços anycast que identifiquem um conjunto de roteadores pertencentes a esses provedores. Assim, os endereços dos provedores confiáveis podem ser citados como endereços intermediários no cabeçalho de roteamento, fazendo com que a rota seja pelos canais desses provedores.

A Figura 58 mostra o formato pré-definido para os endereços anycast, denominado subnet-router anycast address, onde o prefixo de sub-rede indica um link específico.

Figura 58 - Formato do endereço subnet-router anycast address

Os endereços anycast não podem ser associados como endereço de origem de qualquer pacote IPv6, sendo utilizados para roteadores. Este tipo de endereço é sintaticamente o mesmo endereço unicast, tendo os bits do identificador da interface zerados.

4.6.3 Endereço Multicast

Os endereços *Multicast* são utilizados para identificar um grupo de interfaces ou um grupo de nós, mas um pacote enviado a um endereço multicast é enviado para todas as interfaces do grupo. Uma interface pode pertencer a mais de um grupo *multicast*. [61].

Todos os roteadores em uma rede IPv6 reconhecem os endereços multicast, sendo que o prefixo contendo oito bits em nível alto indica um endereço multicast.

Os endereços desse tipo têm o formato mostrado na Figura 59 [54].

Figura 59- Formato do endereço multicast do IPv6

Onde:

- **11111111** - identifica o prefixo, sendo que um endereço iniciado por FF indica ser um endereço multicast.
- **FLAGS** – 4 bits, sendo o formato 000T, com os três primeiros bits reservados e iniciados em zero, e o bit T em 0, indicando um endereço multicast permanente, normalmente atribuído pela autoridade de numeração global da internet, e quando em 1 indicando um endereço temporário.
- **ESCOPO** – 4 bits, limita o escopo dos endereços multicast para nó-local, site-local, link-local, organização local, global, etc, como mostra a Figura 59.
- **ID do GRUPO** – 112 bits, identifica o grupo *multicast* se permanente ou não, dentro do escopo dado.

Exemplo: atribuímos um endereço multicast a um grupo de servidores (TESTE) com um ID de grupo 100 e diversos indicadores de escopo, como mostra a Figura 60.

Figura 60 – Exemplo de endereço multicast

A RFC 2373 definiu diversos endereços multicast, conhecidos como reservados, como mostra a Tabela 5.

Tabela 5 – Endereços multicast reservados

Endereços multicast reservados
FF00:0:0:0:0:0:0:0
FF01:0:0:0:0:0:0:0
FF02:0:0:0:0:0:0:0
FF03:0:0:0:0:0:0:0
FF04:0:0:0:0:0:0:0
FF05:0:0:0:0:0:0:0
FF06:0:0:0:0:0:0:0
FF07:0:0:0:0:0:0:0
FF08:0:0:0:0:0:0:0
FF09:0:0:0:0:0:0:0
FF0A:0:0:0:0:0:0:0
FF0B:0:0:0:0:0:0:0
FF0C:0:0:0:0:0:0:0
FF0D:0:0:0:0:0:0:0
FF0E:0:0:0:0:0:0:0
FF0F:0:0:0:0:0:0:0

Para atribuição de endereços multicast para os nós, utiliza-se os endereços FF01:0:0:0:0:0:1 para atributos locais dos nós e FF02:0:0:0:0:0:1 para atributos locais do link.

4.7 Transmissão de pacotes IPv6 em redes Ethernet

A transmissão de pacotes IPv6 pela mídia de broadcast, por exemplo, nas redes Ethernet, é definida na RFC 2464, tendo como tamanho padrão da MTU, o valor 1500, podendo ser reduzida pelos roteadores ou pela configuração manual. As transmissões com valores de MTU maior de 1500, precisam ser ignoradas em redes Ethernet para a transmissão do pacote IPv6. Durante esta transmissão o formato do pacote decodificado é mostrado na Figura 61.

Figura 61 – Transmissão IPv6 através de redes Ethernet

O valor hexadecimal 86DD no campo de tipo, de um pacote Ethernet indica que o campo de dados contém um cabeçalho e carga útil do IPv6. O enchimento poderá ser utilizado, caso o tamanho mínimo de pacote para Ethernet, não tenha sido alcançado. Um endereço local do link para uma interface Ethernet é formado, acrescentado-se o ID da interface ao prefixo FE80::/64.

4.8 Cenários de Transição do protocolo IPv4 para IPv6

Para que possa ocorrer a transição do protocolo IPv4 para o IPv6, em redes intranet e a transição na Internet, dependemos da compatibilidade da base instalada da rede IPv4. O protocolo IPv6 é capaz de enviar, rotear, ou receber pacotes IPv4, no entanto, os clientes IPv4 não têm condições de tratar os pacotes IPv6.

Para realizar essa transição, poderia ser sugerido que em um determinado dia, todos os sistemas do planeta fariam a atualização do IPv4 para IPv6, porém, com o número de sistemas e o crescimento da internet, essa técnica nem poderia ser imaginada, quanto mais implementada. As especificações contidas na RFC 1933 sugerem dois métodos de transição, ou seja: a pilha IP dupla e o tunelamento do IPv6 por IPv4. Essas duas técnicas podem ser usadas isoladamente ou juntas, para que ocorra a integração gradual de hosts e roteadores IPv6 a uma rede IPv4.

4.8.1 Cenário de Transição – utilização da pilha dupla IPv4/IPv6

A transição entre os protocolos IPv4 e IPv6, utilizando a pilha IP dupla, necessita que um cliente instale as pilhas de protocolo IPv4 e IPv6, possibilitando que o sistema envie e receba pacotes IPv4 e IPv6. Esse nó ou sistema é denominado como nó IPv6/IPv4. Atualmente, vários fornecedores de sistema operacional, oferecem pilhas IPv6 que podem coexistir com as pilhas IPv4, já instaladas. O nó IPv6/IPv4 deve ser configurado com o endereço IPv6 e com o endereço IPv4. Além disso, esses nós precisam entender, se são capazes ou não, de aceitar IPv6 ou somente IPv4.

O protocolo IPv4 define o endereço 127.0.0.1, como sendo o endereço de “loopback”, ou seja, o endereço do próprio nó. A implementação de pilha dupla IPv6/IPv4 pode tratar o endereço IPv6 compatível com IPv4 - ::127.0.0.1 - como

sendo o endereço de “*loopback*” do IPv6. Deste modo, os pacotes com esse endereço devem permanecer dentro do nó.

Outro fator a ser considerado é o registro do recurso de DNS. O tipo de registro de recurso chamado “AAAA” é atribuído aos endereços IPv6, e “A” é atribuído aos endereços IPv4. A implementação da pilha dupla IPv6/IPv4 deve oferecer bibliotecas para trabalhar com os dois tipos de registros DNS.

A Figura 62 mostra um cenário proposto com a implementação da técnica de pilha dupla IPv4/IPv6, para a transição.

Figura 62 – Implementação de pilha dupla IPv4/IPv6

4.8.2 Cenário de transição - Tunelamento de IPv6 sobre IPv4

A infra-estrutura de roteamento do IPv6 será construída com o passar do tempo, e a utilização mais provável, da infra-estrutura de IPv4 ainda funcional, onde poderá ser montada uma infra-estrutura IPv6, e nesse caso, os pacotes IPv6 precisam ser transportados pela rede IPv4. Essa técnica, conhecida como tunelamento, é

definida na RFC 1933. Os *hosts* e roteadores IPv6/IPv4 podem realizar o tunelamento de um datagrama IPv6 dentro dos pacotes IPv4, e assim, serem transportados pela infra-estrutura do IPv4.

Esse tunelamento pode ser usado de várias maneiras:

- **Roteador para roteador:** os roteadores IPv6/IPv4, conectados através da infra-estrutura do IPv4, realizam o tunelamento de pacotes IPv6 entre eles.

- **Host para roteador:** um host IPv6/IPv4 pode fazer o tunelamento para um roteador IPv6/IPv4 intermediário, que possa ser alcançado por meio da rede IPv4.

- **Host para host:** um host IPv6/IPv4 pode fazer o tunelamento para outro host IPv6/IPv4, que possa ser alcançado por uma rede IPv4 para transportar pacotes IPv6.

- **Roteador para host:** um roteador IPv6/IPv4 pode fazer o tunelamento de pacotes IPv6 para o seu destino final.

O processo de tunelamento utilizado é o encapsulamento do pacote IPv6 no IPv4, mostrado na Figura 63.

Figura 63 – Encapsulamento do datagrama IPv6 no pacote IPv4

Um nó como o roteador, que encapsula um datagrama IPv6 dentro de um pacote IPv4, necessita realizar algumas funções mais complexas, por exemplo, identificar quando informar um erro de pacote ICMP muito grande para a origem e determinar como informar os erros ICMP do IPv4 para a origem como um erro ICMP do IPv6.

Foram definidas duas técnicas de tunelamento, diferenciadas, principalmente, na maneira como identificam o endereço da extremidade de tunelamento.

4.8.3 Tunelamento sobre IPv4 configurado ou estático.

Esse mecanismo é utilizado para enviar qualquer pacote IPv6 através de uma região IPv4. A Figura 64 mostra um tunelamento configurado entre uma interface IPv4 no roteador R1 e uma interface IPv4 no roteador R2. Esses dois roteadores estão rodando pilhas IPv6/IPv4.

Para o caso do host IPv6, conectado à região A, enviar um pacote endereçado para o host IPv6, conectado na região B, teremos as seguintes situações ocorrendo:

- O roteador R1 recebe o pacote IPv6, e identifica que deverá encaminhar o pacote por meio de sua interface de tunelamento.
- O roteador R1 encapsula o pacote IPv6 num pacote IPv4. O endereço de origem no cabeçalho IPv4 será o endereço IPv4 da interface de tunelamento local no roteador R1, e o endereço de destino será o endereço IPv4 da interface de tunelamento remoto no roteador R2.
- Um roteador IPv4 na região IPv4, atuando como intermediário, encaminhará o pacote para o roteador R2, baseando-se nas informações do cabeçalho IPv4.
- O roteador R2, então, irá retirar o encapsulamento do pacote, que é o cabeçalho IPv4, e encaminhando-o para o destino, que é um host IPv6.

Figura 64 – Tunelamento de datagrama IPv6 sobre IPv4 configurado

4.8.4 Tunelamento sobre IPv4 automático.

Esse mecanismo é utilizado para encaminhar um pacote IPv6 unicast, com um formato de endereço compatível com IPv4, que é um tipo especial de endereço IPv6. Ele é identificado com um prefixo de 96 bits com todos os valores zeros e um endereço IPv4 nos 32 bits de baixa ordem.

Esses endereços são atribuídos ao nó IPv6/IPv4, que oferece suporte ao tunelamento automático. Os nós que são configurados com endereços compatíveis com IPv4, podem utilizar o endereço inteiro, como sendo seu endereço IPv6, e o endereço IPv4 embutido como seu endereço IPv4. Os roteadores que trabalham com tunelamento automático, devem utilizar as pilhas de protocolo IPv4 e IPv6.

O tunelamento automático procura encaminhar os pacotes para o destino, sendo estabelecido pelo roteador, apenas para o encaminhamento de pacotes IPv6, e após o pacote ter sido encaminhado, o tunelamento termina. A Figura 65 mostra um exemplo de tunelamento IPv4 automático.

Figura 65 – Tunelamento IPv4 automático

4.8.5 Tunelamento IPv6

O tunelamento sobre IPv6 também encapsula os pacotes IP, mas é IPv6 dentro do IPv6, sendo esta técnica utilizada para estabelecer um link virtual entre dois nós IPv6, transmitindo pacotes de dados como carga útil de um pacote IPv6.

O tunelamento sobre IPv6, mostrado na Figura 66 é um mecanismo unidirecional para onde o fluxo de pacote do tunelamento ocorre em uma direção entre os nós de “ponto de entrada” e “ponto de saída” do IPv6. O ponto de entrada pode ser considerado, o que coloca seus próprios pacotes ou de outros, dentro de um pacote, e o ponto de saída é o que retira de dentro deste pacote os seus próprios ou de outros.

Figura 66 – Tunelamento IPv6 unidirecional

A técnica de tunelamento bidirecional é a utilização de dois mecanismos unidirecionais, onde dois túneis são configurados em direções opostas, um ao outro, tendo o nó do ponto de entrada de um tunelamento como nó do ponto de saída do outro tunelamento, mostrado na Figura 67.

Figura 67 – Tunelamento IPv6 bidirecional

4.9 Considerações finais

Neste capítulo foram apresentados conceitos sobre o protocolo IPv6, os formatos dos pacotes, o esquema de endereçamento e atribuição de endereços e cenários a serem utilizados para a migração inicial dos ambientes que utilizam o protocolo IPv4 [62].

O crescimento do tráfego na Internet, estimado em aproximadamente 550 milhões no ano de 2005; as novas tecnologias de redes convergentes; os novos serviços a serem disponibilizados sobre IP, aliados a diminuição de endereços IPv4 livres, permitem imaginar o cenário de transição para suportar estes fatos.

Os estudos e testes experimentais a cada dia vão agregando novas pesquisas e otimizando os resultados no *backbone* IPv6 experimental, e com a entrada de novos parceiros, logo teremos o protocolo IPv6 se tornando um padrão das redes baseadas em IP.

5 MATERIAIS E MÉTODOS

Este capítulo tem por objetivo apresentar os aspectos relevantes do projeto, os materiais e a descrição da metodologia utilizada para a implementação do Laboratório Virtual em um ambiente de sistema operacional de código aberto (Linux), numa rede local com o protocolo de rede IPv6.

O Laboratório Virtual deste trabalho refere-se ao ambiente que proporciona o acompanhamento e o controle de ensaios experimentais com o uso de instrumentação virtual através de uma rede local, tendo como destaques o protocolo IPv6, o sistema operacional Linux e o software LabVIEW® para Linux, da National Instruments.

O resultado deste trabalho apresenta uma rede local, com um microcomputador utilizado para aquisição dos dados do experimento, e também como servidor de aplicação WEB, para a intranet construída com o software Apache. Os clientes acessam o experimento através de um browser, neste caso optamos pelo Netscape.

5.1 Os materiais

Os materiais utilizados neste trabalho relacionam-se ao hardware, ao software e aos mecanismos que possibilitam a comunicação dos computadores e aplicativos numa rede local.

5.1.1 Os equipamentos - Hardware

O servidor utilizado para a aquisição dos dados do ensaio experimental, é um microcomputador modelo PC (Personal Computer) com processador K6-II 500 MHz, 128 Mbytes de memória RAM, disco rígido de 20 Gbytes, placa de rede 10/100, monitor, teclado e mouse e uma placa de aquisição de dados DAQ modelo AT-MIO 16 X, utilizada para a obtenção dos sinais gerados pelo circuito. O software LabVIEW® faz o gerenciamento do Laboratório Virtual, estabelecendo a interface entre os equipamentos de medida e o computador, disponibilizando-as para os usuários.

O circuito eletrônico utilizado como material para o experimento é mostrado na Figura 68, estando identificados os canais de entrada e os componentes envolvidos.

Figura 68 – Circuito eletrônico utilizado no ensaio do Laboratório Virtual

A descrição dos componentes está listada abaixo:

- Fonte de alimentação de 5 Volts
- Placa DAQ PCI-6071E da National Instruments
- Placa SCB-100 da National Instruments
- Amplificador 6825A HP
- Lâmpada 6V-0,1A
- Fototransistor TIL-32
- Resistor de 10K ohms com tolerância de 5%

5.2 Os programas utilizados – Software

As atividades relacionadas à instrumentação virtual possuem tecnologias de software desde a aquisição de dados, a análise e até o compartilhamento das informações, através de uma rede local ou ainda utilizando a infra-estrutura da Internet, para publicação dos resultados. Uma ferramenta com destaque na execução de processos como controle de experimentos é o LabVIEW®.

O LabVIEW® é a ferramenta que atende os requisitos para o controle local de experimentos, proporcionando um ambiente de programação e análise bastante amigável e poderoso (descrito no capítulo 3). Para a implementação do Laboratório Virtual em ambiente aberto, vamos utilizar o LabVIEW® compilado para o Sistema Operacional Linux.

5.2.1 O Sistema Operacional – Linux

O Sistema Operacional Linux está sendo aplicado desde dispositivos de mão a arranjos de computadores de grande capacidade, trabalhando com milhões de transações. O crescimento do Sistema Operacional Linux está relacionado com diversos fatores, entre eles: rapidez, estabilidade, código-fonte completo e gratuito e o fato de que o Projeto GNU – “*General Public License*”, disponibiliza vários utilitários gratuitamente com o código-fonte [63].

O Sistema Operacional Linux, antes utilizado por alguns “excêntricos”, está sendo colocado em vários setores da indústria de computadores, tanto como sistema operacional para servidores, quanto para estações de trabalho. As estimativas atuais colocam a taxa de crescimento do sistema operacional Linux como a mais alta de todos os outros sistemas operacionais combinados.

O Sistema Operacional Linux apresenta um núcleo central – o kernel – e este gerencia os recursos de hardware, possibilitando que mais de um programa estejam em execução num momento, controla a comunicação entre os processos evitando a interferência entre eles e o conseqüente travamento. Para aplicações, os módulos do aplicativo podem ser carregados e executar alguma função específica, por exemplo, a de tornar o computador um servidor de aplicações, ou um servidor WEB, ou ainda um servidor de correio eletrônico, dentre tantas outras possibilidades.

O sistema operacional Linux, tendo o kernel [15] desenvolvido por Linus Torvalds, por volta de 1991, e com programadores espalhados pelo mundo, representa um novo modelo de computação, bem como de distribuição de software, compartilhando idéias e códigos.

Atualmente o Linux se transformou num sistema operacional completo, continuamente sendo desenvolvido e aperfeiçoado, principalmente ao nível do kernel, buscando otimizar o desempenho e a segurança.

Existem muitas razões para a utilização do sistema operacional Linux, algumas delas listadas a seguir:

- **Poder** – transforma computadores antigos, com processadores 486, por exemplo, em sistemas multitarefa, apresentando escalabilidade e confiabilidade a um baixo custo.

- **Desenvolvimento** – possui ferramentas, que aliadas ao projeto GNU, bibliotecas, interfaces gráficas e linguagens orientadas a objeto, tornam o Linux, virtualmente, o sonho dos programadores.
- **Serviços de rede** – através da utilização de aplicativos de servidor, podemos disponibilizar diversos serviços aos usuários de uma rede, como servidor de Banco de Dados, de correio eletrônico, de Internet, entre outros.
- **Portabilidade** – pode ser executado em diversas arquiteturas, entre elas: i386 (arquitetura x86 da Intel, processadores da Cyrix, AMD, IBM, NexGen e outros), Alpha (a CPU DEC Alpha, da Digital), a Arm (a CPU RISC Arm), a mips (o processador MIPS, utilizado nos equipamentos da Silicon Graphics), a ppc (a CPU PowerPC dos computadores Macintosh, da Apple) e a sparc (a CPU Sparc, dos servidores e estações da SUN).
- **Facilidade de depuração** – com o código-fonte pode-se corrigir os problemas apresentados pelo software ou por uma parte específica.
- **Reusabilidade** – utilização de código ou parte do código de uma aplicação em outra, gerando menor tempo de desenvolvimento.
- **Confiabilidade do software** – as correções para o software se tornam mais rápidas, pois vários programadores têm acesso ao código-fonte, podendo apresentar soluções em menor espaço de tempo do que apenas uma equipe de uma empresa comercial.
- **Segurança** – quando detectada uma falha de segurança e reportada, pode ser mais rapidamente corrigida.
- **Instalação de atualização** – quando surge uma atualização para o software, não é necessário esperar pelo fornecedor, ela pode ser instalada imediatamente.

O conceito de código aberto não é novo. Os programadores compartilham códigos desde o início da programação. Com o passar do tempo, os softwares se tornaram proprietários e os usuários recebiam o programa no formato executável, ficando dependente do fornecedor para a correção ou atualização que se fizessem necessárias. O código-fonte aberto tem sido uma tendência, onde diversas empresas

espalhadas ao redor do mundo já o aceitam, e os problemas encontrados e as melhorias, são mais rapidamente implementados.

A motivação para a utilização do Sistema Operacional Linux não fica limitada ao fator econômico, mas agregada a um conjunto de outros fatores, que analisados em conjunto, estão gerando benefícios, e levando as empresas e as instituições acadêmicas, a considerá-lo como uma solução para estações de trabalho e servidores.

5.2.2 O servidor WEB – Apache

A utilização da infra-estrutura de rede da Internet possibilitando o compartilhamento de arquivos de texto, imagens e vídeo, passa efetivamente pelos servidores WEB, que se tornaram parte essencial dos negócios na rede. Da popularidade das empresas na Internet, com páginas cheias de imagens e efeitos sonoros, aos modelos de negócios eletrônicos com lojas virtuais oferecendo serviços e comodidade aos clientes e os serviços bancários são uma amostra do poder e potencial de crescimento das tecnologias que gerenciam os chamados negócios de Internet.

O servidor WEB, como um objeto importante neste conjunto de objetos para a Internet, tem seu desenvolvimento iniciado com o NCSA e atualmente o Apache é o servidor WEB mais popular da Internet [64], sendo extremamente poderoso, escalável e fornecido como pacote em praticamente todas as distribuições do sistema operacional Linux. O “*daemon*” que o Apache [65] instala para fornecer os serviços WEB é o **httpd** (*Hypertext Transport Protocol daemon*).

A configuração do servidor WEB em versões mais antigas era feita por intermédio de três arquivos texto, o **httpd.conf** - para a configuração do “*daemon*” do servidor WEB, o **access.conf** – para controle de acesso e diretório e o **srml.conf** – para configuração dos recursos. Para facilitar e evitar confusão, todos os arquivos foram colocados no **httpd.conf** nas versões mais recentes do Apache.

O servidor WEB também pode ser utilizado nas redes locais, onde o acesso aos documentos pode ser feito através dos navegadores, e este é o modelo que vamos utilizar no projeto do Laboratório Virtual deste trabalho, com o experimento disponibilizado através de uma página HTML, hospedada no servidor Apache.

5.2.3 O protocolo IPv6

Para a construção da rede local de teste do Laboratório Virtual em ambiente aberto, será utilizado o protocolo IPv6, pois a análise das características do cabeçalho (descrito no capítulo 4) mostra possibilidades de configuração de cabeçalhos de extensão, de autenticação e de Qualidade de Serviço, importantes para as aplicações que podem gerar um volume alto de tráfego. O projeto deste trabalho não inclui a instalação do acesso à Internet via protocolo IPv6.

5.3 Etapas pré-estabelecidas para o projeto

O desenvolvimento do trabalho foi escalonado em etapas, sendo que a primeira etapa consiste na seleção de um experimento em instrumentação virtual, para ser utilizado de forma didática e envolvendo protocolo de comunicação entre dispositivos de medida (DAQ), instrumentos (GPIB), microcomputador e uma rede de computadores (protocolo IPv6). O ensaio experimental escolhido está relacionado com o controle de luminosidade de uma lâmpada, oferecendo simplicidade e aplicação em aulas didáticas dos cursos técnicos de eletrônica.

Para a implantação da segunda etapa, que possibilita o controle do experimento via computador, será utilizado o software LabVIEW® compilado para Linux, com o desenvolvimento do VI - instrumento virtual – responsável pela aquisição e controle.

A terceira etapa está relacionada à preparação do ambiente de rede, com a instalação de uma rede com o sistema operacional Linux e a configuração do protocolo IPv6.

A quarta etapa consiste na implementação do servidor WEB, para publicação dos dados através da rede local. O diagrama de blocos na Figura 69 apresenta as etapas a serem cumpridas até a conclusão deste trabalho.

Figura 69 – Etapas a serem desenvolvidas para a realização deste trabalho

5.4 Metodologia adotada

A arquitetura cliente/servidor possibilita a implementação de sistemas de controle em redes locais. O computador a ser utilizado como servidor, é colocado próximo do experimento, e para a comunicação com os instrumentos e o experimento em si, utiliza interfaces como porta paralela, serial ou GPIB.

A transferência entre o servidor e os clientes necessita de um protocolo de comunicação para a rede. Entre os vários protocolos para o transporte de dados em rede, o UDP (*User Datagram Protocol*) – um protocolo da camada de transporte da pilha de protocolo TCP/IP, não orientado a conexão - e o TCP (*Transport Control Protocol*) – um protocolo da camada de transporte da pilha TCP/IP, orientado a conexão, e ainda o protocolo IP (*Internet Protocol*) – um protocolo da camada de rede da pilha de protocolo TCP/IP, são os mais utilizados.

Neste projeto serão utilizados o TCP e o IP, para a transferência dos dados, sendo o TCP para o transporte, por garantir a entrega dos dados, embora mais lento que o UDP. A rede implantada utiliza o protocolo IP com a versão seis, devido a características de segurança e autenticação, qualidade de serviço (QoS), que podem ser configuradas no protocolo, permitindo aproximar a rede local de uma rede determinística.

Para a monitoração e atuação no experimento, incluindo os sensores, instrumentos virtuais (VI) serão implementados em LabVIEW®. O VI de controle do experimento, neste projeto chamado de servidor do experimento, irá controlar a comunicação entre os equipamentos envolvidos, recebendo e enviando as informações (dados) para os ensaios.

O software LabVIEW® possui ferramentas de publicação em servidores WEB, através de páginas HTML. Na versão 6.1 do LabVIEW®, utilizada neste projeto, o protocolo de rede suportado é o IP versão quatro, não tendo ainda suporte ao protocolo IPv6. Esta característica impossibilita a utilização da funcionalidade dos blocos de comunicação TCP/IP do LabVIEW®, para construir a parte do software cliente. A solução adotada será colocar um software de servidor WEB, no caso o Apache, instalado no mesmo computador do servidor de experimento e através da rede local com o protocolo IPv6 implementado no sistema operacional Linux, que são escopos deste trabalho, disponibilizar acesso ao experimento para os clientes.

O sistema proposto é mostrado na Figura 70, onde é possível identificar os instrumentos, o servidor e seus módulos e o cliente, com a comunicação através da rede local IPv6.

Figura 70– Estrutura do sistema de comunicação e aquisição de dados proposto para o projeto

A Figura 71 mostra a metodologia de desenvolvimento utilizada para este projeto.

Figura 71 – Metodologia de desenvolvimento adotada para o projeto

5.5 Considerações Finais

Este capítulo descreveu os materiais e a metodologia utilizada para o desenvolvimento deste trabalho, mostrando a importância e justificando as escolhas das tecnologias. Ao final encontra-se o diagrama de blocos ilustrando as etapas do projeto e uma breve descrição das atividades realizadas em cada etapa.

O próximo capítulo apresentará os resultados obtidos e as conclusões sobre as atividades realizadas em cada etapa.

6 RESULTADOS EXPERIMENTAIS

Neste capítulo são descritas as etapas e os resultados obtidos na implementação do experimento, as características do circuito do instrumento virtual desenvolvido.

6.1 Ensaio experimental

Neste trabalho foi utilizado um circuito eletrônico elaborado para permitir a variação da luminosidade de uma lâmpada através do controle da corrente que circula por um fototransistor. Foi construído um arranjo com um resistor, um fototransistor e uma lâmpada.

O objetivo deste experimento é possibilitar ao usuário um ambiente instrumental operacional para o estudo e controle de luminosidade de uma lâmpada.

A radiação eletromagnética é formada por fótons que possuem associado uma energia luminosa, que se caracteriza pelo comprimento de onda, segundo a equação:

$$E = hf = \frac{hc}{\lambda} \quad (1)$$

Onde:

E - energia do fóton

h – constante de Planck

c – velocidade da luz ($3 \cdot 10^8$ m/s)

λ - comprimento de onda

A energia de um fóton é maior quanto menor o comprimento de onda, pois segundo a equação, o valor constante está no numerador, ficando dependente apenas do comprimento de onda.

A Figura 72 mostra o espectro eletromagnético com os comprimentos de onda para cada tipo de radiação.

Figura 72 – Espectro eletromagnético com as faixas de comprimento de ondas

A radiação eletromagnética é dividida segundo o comprimento de onda, tendo algumas fontes de sinal definidas.

- **Ondas de rádio** – geradas por circuitos osciladores LC e utilizadas nas comunicações.
- **Microondas** – estão na faixa de 1 mm a 30 cm e são adequadas para sistemas de radar, navegação aérea e para estudos das propriedades da matéria.
- **Infravermelho** – também chamadas de ondas térmicas sendo produzidas principalmente por corpos quentes e podem ser absorvidas pela maioria dos materiais, sendo a energia absorvida na forma de calor. Estão na faixa de 1 mm a $4 \cdot 10^{-7}$ m.

- **Ultravioleta** – produzida principalmente pelo Sol, está na faixa de 6.10^{-8} a $3,8.10^{-7}$ m.
- **Raios X e Gama** – são ondas de grande energia e que podem causar danos à estrutura aos tecidos humanos.

A luz visível, percebida pelo olho humano, está na faixa dos comprimentos de onda entre 390 e 780 nm, possuindo cores diferentes para os diferentes comprimentos de onda [66].

A Optoeletrônica trabalha, principalmente, na parte do espectro eletromagnético correspondente à parte visível e nas proximidades do infravermelho.

Os dispositivos optoeletrônicos básicos são:

- **Dispositivos emissores:** emitem luz ao serem ativados por corrente elétrica, portanto são dispositivos que transformam energia elétrica em energia luminosa, por exemplo, os diodos LED e o LASER.
- **Dispositivos detetores:** geram um sinal elétrico ao serem iluminados, portanto, transformam a energia elétrica em energia luminosa.
- **Dispositivos fotocondutores:** conduzem a radiação luminosa de um emissor para um receptor, não produzindo transformação de energia.

6.2 Diodo

Uma junção PN é construída com a fusão de um material semicondutor dopado com um excesso de elétrons, chamado tipo N, e com um material semicondutor dopado com uma deficiência de elétrons (definida como lacunas), chamado tipo P. Quando os dois materiais são colocados juntos, uma reorganização da carga na vizinhança da junção causa o aparecimento de uma barreira de potencial. Para que possa existir condução de corrente, os portadores majoritários devem ultrapassar o potencial desta barreira. A amplitude deste potencial é dependente do material semicondutor e para o silício o valor é de 0,6 Volts.

Para um diodo, a corrente I está relacionada com a tensão V pela equação:

$$I = I_0 (e^{V/\eta kT} - 1) \quad (2)$$

Onde:

I_0 – é a corrente reversa na junção;

V – É A TENSÃO APLICADA;

k – é a constante de Boltzman;

η - é propriedade do material da junção

T – temperatura absoluta.

A Figura 73 mostra as curvas características de diodos para diferentes materiais.

Figura 73 – Curva característica de diodos

Os diodos são sensíveis à luz. A corrente reversa de saturação depende da densidade dos elétrons livres e lacunas, e para o fotodiodo esta corrente I_0 é chamada de corrente “escura”. A luz brilhando sobre uma junção cria pares de elétrons-lacunas livres. A tensão reversa aplicada ao diodo faz com que os portadores livres atravessem a junção, resultando na fotocorrente. A amplitude depende da intensidade da luz brilhando sobre a região da junção. Os fotodiodos são construídos para otimizar este efeito. A Figura 74 mostra as curvas características de um fotodiodo, onde observamos o efeito da ausência de luz e o efeito do aumento da intensidade de luz na junção.

Figura 74 – Curva características do fotodiodo

6.3 Dispositivos Fotodetetores

Os componentes fotodetetores variam algum parâmetro elétrico em função da luz. Num dispositivo construído com material semicondutor, tendo a luz incidindo sobre ele, irá ocorrer a geração de pares de elétrons e lacunas. Os dispositivos fotodetetores podem ser construídos como fotoresistência, um fotodiodo ou ainda um fototransistor.

6.3.1 Fotoresistência

Uma fotoresistência é um material semicondutor cuja resistência varia em função da iluminação, reduzindo o valor da resistência na presença de raios luminosos, sendo chamada de resistência dependente da luz – LDR – Light dependent resistor.

Com a incidência de luz sobre o fotoresistor, o valor da resistência diminui, logo, o fotoresistor quando iluminado possui um valor baixo de resistência.

A Figura 75 mostra um material semicondutor sendo iluminado e a geração de pares de elétrons e lacunas.

Figura 75 – Uma junção PN sendo iluminada e a geração de pares de elétrons e lacunas

Ao retirarmos a incidência de luz sobre o material semicondutor, os portadores fotogerados anteriormente se recombinam e a resistência será maior, como mostra a Figura 76.

Figura 76 – Uma junção PN não iluminada e o efeito da recombinação

6.3.2 Fotodiodo

Os fotodiodos são diodos de junção PN, cujas características elétricas dependem da quantidade de luz que incide sobre a junção.

A Figura 77 mostra o símbolo de um fotodiodo e as curvas características da corrente em função da tensão, onde se observa que o efeito fundamental é a geração de pares de elétrons e lacunas devido à energia luminosa, na região de polarização inversa.

Figura 77 – Símbolo e curva característica de um fotodiodo

Para caracterizar um fotodiodo são definidos os termos: corrente escura – é a corrente reversa quando não existe incidência de luz, e a sensibilidade ao aumento de intensidade de polarização na região reversa por unidade de intensidade de luz, expressa em W/cm^2 .

O fotodiodo, quando polarizado na região de condução direta, se comporta como um diodo comum, conduzindo pequenas correntes e, se fabricado com material semicondutor silício, a queda de tensão será de aproximadamente 0,7 Volts.

6.3.3 Fototransistor

O fototransistor é um transistor bipolar sensível à luz. A radiação luminosa incide sobre a junção coletor-base, o que provoca a geração de pares de elétrons e lacunas, e com isto a corrente elétrica.

O fototransistor opera, em geral, sem o terminal de base, possuindo maior sensibilidade do que um fotodiodo, pois uma pequena corrente fotogerada, é multiplicada pelo ganho do transistor.

A Figura 78 mostra o símbolo do fototransistor e as curvas características da corrente em função da tensão, tendo a corrente de base substituída pela potência da radiação luminosa.

Figura 78 – Símbolo e curvas características de um fototransistor

6.4 Primeira etapa – Seleção do ensaio experimental

O ensaio experimental escolhido para testes da infra-estrutura do Laboratório Virtual utilizando ambiente aberto, consiste em estudar a emissão de luz visível irradiada por um corpo (no experimento utilizamos uma lâmpada), e o controle de sua intensidade luminosa é realizado através de um computador remoto, tendo aplicações práticas em atividades com sensores e fotometria. [65]

A luz visível é uma radiação eletromagnética com comprimento de onda entre 390 e 780 nanômetros e pode ser detectada pelo olho humano. Para estudar as características de emissão luminosa de um corpo conforme a variação de uma grandeza física incidente sobre este corpo, utilizamos um, cuja principal propriedade se encontra na variação de sua resistência elétrica por fatores notáveis. Para determinada intensidade de radiação visível incidente sobre o componente, este adquire uma resistência intrínseca.

No ensaio experimental, a tensão sobre os terminais da lâmpada é variada e com isto a variação de luminosidade, que é então detectada pelo foto-transistor. Para o estudo desejado foi implementado um sistema mostrado em blocos na Figura 79, onde se pode controlar a tensão aplicada aos terminais de uma lâmpada através de um computador remoto, instalado na rede local. O sistema consiste em um aplicativo computacional, desenvolvido como um VI - instrumento virtual, utilizando o software LabVIEW®, e que pode controlar a tensão de saída de uma placa de aquisição de sinais em um outro computador, denominado no projeto como servidor do experimento. Nas saídas da placa de aquisição de sinais (placa DAQ) estão

conectados os terminais de uma lâmpada. Utilizamos um fototransistor, ligado em série com um resistor de 10K ohms, colocado de frente para a lâmpada e polarizado por uma tensão constante, de tal forma que a luz visível emitida neste, influencia na passagem de corrente pelo componente. A corrente que circula no fototransistor é calculada através da relação entre a tensão no resistor (obtida com a aquisição realizada pela placa de aquisição de dados) e sua resistência, utilizando-se a lei de Ohm.

Figura 79 – Diagrama de blocos do ensaio experimental

O circuito eletrônico utilizado para o experimento é apresentado na Figura 80.

Figura 80 – Circuito eletrônico utilizado no ensaio experimental

6.5 Segunda etapa – Implementação do Instrumento Virtual

Para a implementação do instrumento virtual, utilizando o LabVIEW® para controle do experimento foram desenvolvidos dois VIs, o VI Servidor – para

funcionamento no servidor de experimento e o VI Cliente – para funcionamento no computador cliente. Neste primeiro teste do laboratório remoto em ambiente aberto, utilizamos o sistema operacional Linux, num ambiente de rede com o protocolo IPv4 sendo o protocolo da camada de rede, e com isto as funcionalidades de comunicação TCP/IP do software LabVIEW® puderam ser implementadas no projeto.

6.5.1 Implementação do VI do Servidor

A Figura 81 mostra o diagrama de blocos (programação em linguagem gráfica) implementado no servidor. A conexão TCP aguarda por um processo de um cliente solicitando conexão. Nesta conexão o cliente envia os parâmetros que deseja para a realização do experimento e que estão disponíveis no painel frontal apresentado ao cliente. Estes parâmetros podem ser:

- 1 – Controle da tensão de Off-set;
- 2 - Controle do valor da frequência;
- 3 – Controle da amplitude;
- 4 – Interrupção da execução (STOP);
- 5- Seleção do experimento (foto-transistor, LDR, termopar).

O VI implementado no servidor também é responsável pela estrutura de seleção do experimento, pelo envio do sinal medido pela placa de aquisição de dados para o cliente e também por fechar a conexão.

Figura 81 – Diagrama de blocos do VI implementado no servidor

O VI do servidor do experimento é composto por uma série sequencial de um Sub-VI chamado “DataReceive”, que tem a função de receber os dados de controle do sinal do computador cliente. Este Sub-VI contém um gerador de sinal senoidal, onde os controles de amplitude, frequência e “off-set” são comandados pelo cliente. A estrutura condicional é utilizada para seleção de um experimento, podendo ser um fototransistor, LDR ou um termopar. A estrutura sequencial, composta por duas partes, é responsável por enviar o sinal ao atuador e a segunda mede o sinal do sensor analisado, através da placa de aquisição de dados.

O painel frontal disponível no servidor, apresentado na Figura 82 mostra a porta de comunicação, a frequência do sinal, a amplitude do sinal, o off-set do sinal, o sinal aplicado no elemento atuador e a sinal medido no sensor analisado.

Figura 82 – Painel frontal disponível no Servidor do Experimento

O diagrama de blocos chamado Sub-VI DataReceive é mostrado na Figura 83.

Figura 83 – Diagrama de blocos do Sub-VI DataReceive

6.5.2 Implementação do VI Cliente

A Figura 84 mostra o diagrama de blocos (programação em linguagem gráfica) implementado no cliente. A solicitação de conexão TCP é enviada para o servidor, iniciando a comunicação. Nesta conexão, o cliente envia os parâmetros que deseja para a realização do experimento e que estão disponíveis no painel frontal,

apresentado ao cliente. Os blocos numerados didaticamente de I a V, são Sub-VI chamado *DATA_Send* que recebem os parâmetros de configuração desejados pelo cliente, ou seja:

- I – Envia a tensão contínua desejada para o servidor (tensão offset)
- II – Envia a frequência do sinal AC
- III – Envia a amplitude do sinal AC
- IV – Envia um dado para o servidor indicando que o cliente se desconectou com o servidor. O valor lógico zero indica que houve desconexão e o valor lógico um indica que ele ainda está conectado.
- V – Envia um dado indicando o tipo de experimento a ser realizado.

Figura 84 – Diagrama de blocos do VI implementado no cliente

O VI do cliente do experimento ainda possui dois conjuntos responsáveis pela recepção de sinais, sendo um para os sinais amostrados (medidos) e o outro para os sinais aplicados.

O diagrama de blocos chamado Sub-VI *DATA_Send* é mostrado na Figura 85.

Figura 85 – Diagrama de blocos do Sub VI DATASend

O painel frontal disponível no cliente, apresentado na Figura 86 mostra os sinais de controle aplicados e os sinais amostrados. Possui ainda indicadores que possibilitam controlar o sinal a ser aplicado no experimento, selecionar o tipo de experimento a ser realizado e a indicação da porta e do endereço de conexão.

Figura 86 - Painel frontal disponível no cliente

6.6 Conclusões parciais

O experimento realizado nesta etapa com o Laboratório Remoto, utilizando a infra-estrutura de ambiente aberto, e o protocolo IPv4, como protocolo de rede, permitiram o desenvolvimento e os testes iniciais dos VIs com o LabVIEW® rodando em sistema operacional Linux. Na próxima etapa deste projeto, será implementada a infra-estrutura de rede com o protocolo IPv6 e através das ferramentas de publicação na WEB do LabVIEW®, o cliente terá acesso através de um navegador endereçando um servidor Apache.

7 RESULTADOS EXPERIMENTAIS – AMBIENTE DE REDE

Neste capítulo são descritos as etapas e os resultados obtidos na implementação da infra-estrutura de rede para o laboratório Virtual em ambiente aberto utilizando o protocolo IPv6.

7.1 Implementação do Ambiente de rede

Para a implementação da infra-estrutura de rede do Laboratório Virtual utilizando sistema operacional aberto, a distribuição Red Hat versão 8.0 [67] é a escolhida para ser instalada no servidor, que irá receber e enviar os dados do experimento. Os computadores que serão configurados como clientes da rede experimental, também será instalado o Red Hat versão 8.0. A escolha desta distribuição se deve ao fato de ser uma das mais utilizadas, além de servir de base para outras.

O sistema operacional Linux é formado por um kernel e módulos que são compilados quando necessários, o que permite a configuração de quais módulos o kernel dará suporte. Para que o kernel do sistema operacional Linux possa trabalhar com o protocolo IPv6, devemos carregar o módulo referente ao protocolo, habilitando o kernel a tratar os cabeçalhos dos pacotes na rede IPv6 [62].

Para verificar se numa distribuição Linux, em particular, o kernel oferece suporte ao protocolo IPv6, deve-se localizar o arquivo *if_inet6* que é instalado no diretório /proc/net. Caso ele exista, o kernel já possui suporte ao protocolo IPv6, bastando carregar o módulo. Para os casos em que o kernel não tenha suporte ao protocolo IPv6, deve-se obter os arquivos-fonte do pacote “*net-tools*” e recompilá-lo. Esses arquivos podem ser obtidos, por exemplo, no endereço de Internet: www.tazenda.demon.co.uk/phil/net-tools/.

As distribuições mais recentes trazem o pacote “*net-tools*” compilado. Para verificar sua existência, utilize o programa utilitário `route -help`, que se encontra no diretório /sbin. Caso a resposta seja uma lista e o texto “inet6 (IPv6)” apareça, então o kernel está pronto.

Na distribuição do Red Hat 8.0 o arquivo **modules.conf** instalado no diretório /etc fornece informações referentes aos módulos carregados.

7.2.1 Software de Servidor WEB – Apache

Na primeira implementação do experimento inicial, foi desenvolvido um VI para que o controle fosse realizado pelo cliente, em uma rede com o protocolo IPV4, utilizando as funcionalidades do TCP/IP, oferecidas pelo LabVIEW®, para testes em um ambiente com o sistema operacional Linux. Como o objetivo desta dissertação está ligado ao protocolo IPv6, foi instalado um software de servidor WEB com funções de controle de acesso e disponibilização do experimento na rede local.

7.2.1.1 Instalação do software Apache

A instalação do servidor Apache pode ser feita com os códigos que, em geral, acompanham as distribuições Linux, porém, o ideal é procurar os códigos fontes diretamente do “site” do Apache [64] e compilar. O arquivo com o formato `httpd-2.0.xx.tar.gz`, onde `xx` indica a última versão disponível, deve ser copiado para um diretório, na estrutura de diretórios do computador que será configurado como o servidor Apache. Como sugestão, pode-se criar o diretório `httpd-2.0.xx`.

Como observação, o processo de instalação e configuração do servidor Apache deve ser realizado pelo administrador do sistema, ou seja, o **root**. No sistema operacional Linux, o caractere `#` indica que o usuário *logado* no sistema é o **root**.

Para descompactar o arquivo `httpd-2.0.xx.tar.gz`, utilizamos os comandos abaixo:

```
# gzip -d httpd-2.0.xx.tar.gz que irá gerar o arquivo .tar
```

```
# tar xvf httpd-2.0.xx.tar.gz
```

Para compilar e instalar o Apache, executamos a sequência de comandos a seguir:

```
# cd httpd-2.0.xx (este diretório contém os códigos do Apache)
```

```
# ./configure --prefix=/usr/local/apache (indica o diretório de instalação do Apache).
```

```
# make
```

```
# make install
```

Para realizar os testes iniciais do servidor Apache, deve ser inicializado o serviço **apachectl** e através de um navegador instalado no mesmo computador, acessar o endereço `http://127.0.0.1`, ou o endereço `http://::1/128` que corresponde ao endereço da interface de “*loopback*”. Como resultado de uma instalação correta deverá aparecer uma tela de boas vindas.

Para este teste, executamos o comando descrito abaixo:

/usr/local/apache/bin/apachectl start, que inicializa o servidor Apache.

O próximo procedimento será configurar o servidor Apache, mas antes, como sugestão, os arquivos de instalação devem ser apagados e assim liberar espaço no disco. Os comandos abaixo realizam esta tarefa.

rm -rf http-2.0.xx

rm -rf http-2.0.xx.tar.gz

7.2.1.2 Configuração do servidor Apache

Para a configuração do servidor Apache, o diretório padrão (raiz) e um diretório para os arquivos de *scripts* CGI – Common Graphics Interface, devem ser definidos.

A sequência de comandos abaixo, cria a estrutura dos diretórios necessários e as permissões para leitura e escrita nestes diretórios.

mkdir /home/www (cria o diretório raiz do servidor)

mkdir /home/www/cgi-bin (cria o diretório para os arquivos de scripts)

chmod 751 /home/www

chmod 751 /home/www/cgi-bin

Estas duas linhas configuram as permissões necessárias, ou seja, acesso total para usuário, leitura e execução para grupo e execução para outros.

O arquivo de configuração do servidor Apache é o arquivo **httpd.conf**, no formato texto, localizado no diretório **/usr/local/apache/conf** [49]. Este arquivo, organizado em seções, possibilita a configuração dos seguintes itens:

- **Nome do Site** – configura-se o hostname do Servidor, podendo ter as páginas acessadas pelo endereço de “*loopback*” (127.0.0.1 ou ::1/128).

- **Administrador do Site** – configura-se o endereço de correio eletrônico do administrador do site, para a geração automática de páginas, como por exemplo, mensagens de erros.
- **Diretório raiz** – configura o diretório padrão para a instalação do site.
- **Diretório para scripts CGI** – configura o diretório onde estão os arquivos com os scripts CGI.
- **Extensão dos arquivos CGI** – configura quais as extensões de arquivos serão aceitas como scripts CGI.
- **Extensão dos arquivos SSI** – Server Side Include – configurar para que os arquivos com a extensão .shtml sejam considerados como arquivos do tipo HTML.

Para configurar o arquivo `httpd.conf`, utiliza-se um editor de texto disponível no sistema operacional Linux e a partir de um arquivo padrão, que é instalado durante o processo de instalação do Apache, são alteradas as opções de acordo com o características desejadas para o servidor. A configuração e a manutenção do servidor WEB se torna assim, um processo fácil execução na administração do Laboratório Virtual.

Para facilitar a compreensão do procedimento utilizado na implementação do servidor Apache neste projeto, escrevemos a linha de comando, colocada durante o processo de instalação e, a seguir, a linha alterada para nossos propósitos.

Devemos observar que o sistema operacional Linux considera o caractere #, colocado em linhas nos arquivos texto, é considerado um comentário, que significa que no caso de uma linha de comando, a mesma não será executada. Nos procedimentos de configuração descritos a seguir, em alguns casos, apenas foi retirado o caractere de comentário.

Configuração do nome do site:

```
# ServerName host.domain.name:80 alterada para:
ServerName www.saturno.tk:80
```

Configuração do endereço de correio do administrador:

```
# ServerAdmin you@your.address alterada para:
ServerAdmin varella@poli.usp.br
```

Configuração do diretório raiz para o site:

DocumentRoot "/usr/local/apache/htdocs" alterada para:

DocumentRoot "/home/www"

<Directory "/usr/local/apache/htdocs" > alterada para:

<Directory "/home/www">

Configuração dos nomes dos arquivos padrões:

DirectoryIndex index.html index.html.var alterada para:

DirectoryIndex index.html index.html.var index.shtml

Configuração do diretório para os arquivos de scripts CGI:

ScriptAlias /cgi-bin"/usr/local/apache/cgi-bin/" alterada para:

ScriptAlias /cgi-bin"/home/www/cgi-bin/"

<Directory "/usr/local/apache/cgi-bin"> alterada para:

<Directory "/home/www/cgi-bin">

Configuração da extensão dos arquivos CGI:

#AddHandler cgi-script .cgi alterada para:

AddHandler cgi-script .cgi

Configuração da extensão dos arquivos SSI:

#AddType text/html .shtml

#AddOutputFilter INCLUDES .shtml alteradas para:

AddType text/html .shtml

AddOutputFilter INCLUDES .shtml

8 CONCLUSÕES E PERSPECTIVAS FUTURAS

Foi montada a infra-estrutura de um ambiente, de sistema operacional aberto, para a aquisição de dados em redes remotas e com isto promover a cooperação e o compartilhamento de instrumentos, notadamente nas áreas acadêmica e de pesquisa, sem prejuízo da utilização nas outras áreas do conhecimento.

Foram apresentados os conceitos básicos sobre o protocolo IPv6, descrevendo suas características, funcionalidades e os possíveis cenários de transição para a implementação de uma rede de computadores que permita o acesso a um Laboratório remoto.

O projeto implementado consiste num ambiente de rede, com o sistema operacional Linux, o protocolo IPv6, responsável pela estrutura de endereçamento da rede local e um sistema de aquisição, controle e gerenciamento, responsável pela interligação de instrumentos de medida, placa de aquisição de dados de um experimento didático. Este controle pode ser realizado em rede local utilizando acesso via browser ao servidor WEB.

O experimento didático foi desenvolvido em ambiente de instrumentação virtual, baseado na ferramenta LabView® para Linux, da National Instruments, para controle do aparato experimental.

Como perspectiva de atividades futuras dando continuidade aos resultados obtidos neste trabalho, pretende-se elaborar experimentos relacionados aos conteúdos programáticos dos cursos: de eletrônica, telecomunicações e automação industrial e disponibilizá-los aos alunos das escolas técnicas contribuindo, desta forma, com a diminuição dos custos envolvidos na construção de laboratórios e possibilitando a educação continuada, diminuindo as distâncias entre os alunos e professores na transmissão e aquisição dos conhecimentos.

Como resultado deste trabalho é disponibilizado um Laboratório Virtual em ambiente aberto, que possibilita o controle de um ensaio experimental, através de uma rede local.

Em termos específicos de extensão dos resultados obtidos nesta pesquisa são propostos os seguintes estudos:

1. Implementação do acesso ao experimento através da Internet, utilizando o backbone experimental com o protocolo IPv6 nativo.
2. Implementação de controle de acesso aos experimentos, que devem ser oferecidos por curso e aluno matriculado, fornecendo teoria básica do assunto a ser desenvolvido experimentalmente, a resolução das atividades práticas referentes e um sistema de inserção e correção de defeitos.
3. Implementação do Laboratório Virtual em redes de alta velocidade como a Internet2, disponibilizando novos recursos, como transmissão de vídeo e áudio.

REFERÊNCIAS BIBLIOGRÁFICAS

- [1] MEC, Ministério da Educação.
Disponível em: www.mec.gov.br. Acesso em 02/07/2003.
- [2] SEMTEC, Secretaria da Educação Média e Tecnológica.
Disponível em: www.mec.gov.br/semtec. Acesso em: 02/07/2003.
- [3] ARPAIA, P., BACCIGALUPI, A., CENNAMO, F. e DAPONTE, P., "A remote measurement laboratory for educational experiments", *Measurement*, Vol. 21, Nº 4, p. 157-169, 1997.
- [4] WALLER, J. C. e FOSTER, N., "Training via the web: a virtual instrument", *Computers & Education*, Vol. 35, p. 161-167, 2000.
- [5] TANER, A H; BRIGNELL, J. E.; "Virtual Instrumentation and intelligent sensors", *Sensors and ActuatorsA*, vol.61, pp. 427-430,1997
- [6] GOLDBERG, H.; "What is Virtual Instrumentation?", *IEEE Instrumentation & Measurement Magazine*, pp. 10-13, December, 2000.
- [7] BECK, T.; "Current trends in the design of automotive electronic systems Design", *Automation and Test Conference*, 2001.
- [8] HOON, S. P.; "Conducting Experiments over the Internet". *Instrumentation Newsletter*, National Instruments, third quarter 1999.
- [9] Massachussets Institute of Technology – MIT
Disponível em:
<http://www.media.mit.edu/wearables/mithril/context/topic1.html>
Acesso em: 12/10/2002.
- [10] NIPPERT, C. R.; "On Line Experiments – The Results of the Online Widener Laboratories", *Proceeding of the 2002 ASEA/IEEE Annual Conference*, Boston, MA, EUA.
- [11] ESCHE, S. K. & CHASSAPIS, C.; "An Internet -based remote-access approach to undergraduate laboratory education". *Proceedings of the 1998 Fall Regional Conference of the Middle Atlantic Section of ASEE*, pp. 108-113.
- [12] BERTOCCO, M.; FERRARIS, F.; OFFELI, C.; PARVIS, M.; "A Client-Server Architecture for Distributed Systems", *IEEE Transaction on Instrumentation and Measurement*, vol. 47 no. 5, October, 1998.

- [13] STEGAWSKI, M. A; SCHAUMANN, R., "A New Virtual-Instrumentation-Based Experimenting Environment for Undergraduate Laboratories with Application in Research and Manufacturing", IEEE Transaction on Instruments and Measurement, vol. 46, no. 6, December, 1998.
- [14] PALOP, J. M.; TERUEL, J. A., "Virtual Work Bench Electronic Instrumentation Teaching", IEEE Transaction on Education, vol. 34, pp.1-15, 2000.
- [15] KERNEL.ORG. Disponibiliza o kernel do Linux. Disponível em: <http://www.kernel.org> . Acesso em: 16/12/2002
- [16] KLEIN, A., "Teleteaching scenarios for high bandwidth networks", Computer Networks and ISDN Systems", N° 30, p. 1707-1716, 1998.
- [17] PASSERINI, K., GRANGER, M. J., "A developmental model for distance learning using the Internet", Computers & Education, Vol. 34, p. 1-15, 2000.
- [18] IPv6. Disponibiliza informações sobre o protocolo IPv6. Disponível em: <http://www.ipv6.org>. Acesso em 16/12/2002.
- [19] SIM, site do Grupo de Sensores Integráveis e Microsistemas. Disponível em: <http://sim.lme.usp.br>. Acesso em: 12/06/2003.
- [20] NATIONAL INSTRUMENTS. Apresenta requisitos necessários para o kernel do Linux e a utilização do LabVIEW® neste ambiente. Disponível em : <http://www.ni.com/linux/release.htm> . Acesso em: 29/06/2003.
- [21] CRISTALDI, L.; FERRERO, A ; PIURI, V., "Programmable Instruments, Virtual Instruments and Distributed Systems: What is really useful, inovative and Technically Sound?", IEEE Instrumentation & Measurement Magazine, September, 1999.
- [22] BORGES, A; FERNANDEZ, R.; PERES_LISBOA, M.; MORI, A ; RAMIREZ-FERNANDEZ, F. J., "Virtual Laboratory Associated to Intelligent Instrumentation", IES 2000 – Internet Education Science, Ucrânia, 2000.
- [23] MARINO, P.; NOGUEIRA, J.; HERNANDEZ, H., "Laboratory of virtual instrumentation for industrial eletrronics", Proceedings of IEEE International Conference on Industrial Technology, vol. 2, 2000.
- [24] WANG, C.; GAO, R., "A Virtual Instrumentation System for Integrated Bearing Condition Monitoring", IEEE Transaction on Instrumentation and Measurement, vol 49, no. 2, April, 2000.

- [25] KUROSE, J.; ROSS, K. Redes de Computadores e a Internet. São Paulo: Addison Wesley, 2003. 548 p.
- [26] TANEMBAUM, A., "Redes de Computadores" – 3ª edição, Rio de Janeiro, Editora Campus, 1996.
- [27] COMER, D., "Internetworking with TCP/IP- Principles, Protocols and Architecture", 3a. edição, 1997.
- [28] PATON, B., "Virtual Laser Lab – The Future of Distance Learning is Here!", Instrumentation Newsletter, National Instruments, vol.10, no. 4, 1998.
- [29] OVERSTREET, J.W.; TZES, A., "Internet-based client/server virtual instrument designs for real-time remote-access control engineering laboratory", American Control Conference, vol. 2, 1999, pp. 1472-1476.
- [30] SPOELDER, H. J. W., "Virtual Instrumentation and Virtual Environments", IEEE Instrumentation & Measurement Magazine, September, 1999.
- [31] BISHOP, R. H., "LabVIEW Student Edition 6.i", NJ, Prentice Hall, 2001.
- [32] NATIONAL INSTRUMENTS. Apresenta requisitos necessários para a utilização do LabVIEW®.
Disponível em : <http://www.ni.com/linux/release.htm>
Acesso em: 16/12/2002.
- [33] NATIONAL INSTRUMENTS, "Integration the Internet into Your measurement system. DataSocket technical Overview", 2000
Disponível em : <http://www.ni.com/linux/release.htm>
Acesso em: 18/01/2003.
- [34] Nierle HomePage. Internet Protocol Version 6 - June 13, 1996.
Disponível em: <http://www.stl.nps.navy.mil/~jenierle/chapt6.html>
Acesso em: 12/12/2002
- [35] RFC 2373 - IP Version 6 Addressing Architecture. HINDEN, R., DEERING, S. July 1998. Disponível em: <http://www.ietf.org/rfc/rfc2373.txt> .
Acesso em: 10/10/2002
- [36] RFC 2460 – Internet Protocol Version seis (IPv6) Specification. DEERING, S., HINDEN, R., December 1998.
Disponível em: <http://www.ietf.org/rfc/rfc2460.txt> . Acesso em: 10/10/2002
- [37] RFC 1886 – DNS Extensions to Support IPv6. THOMSON, S., HUITEMA, C., December 1995.

Disponível em: <http://www.ietf.org/rfc/rfc1886.txt> . Acesso em: 10/10/2002

[38] RFC 2471 – IPv6 Testing Address Allocation. HINDEN, R, FINK, R., POSTEL, J. December 1998. Disponível: <http://www.ietf.org/rfc/rfc2471.txt> Acesso em: 10/10/2002.

[39] RFC 2893 – Transition Mechanisms for IPv6 Hosts and Routers. GILLIGAN, R., NORDMARK, E., August 2000. Disponível em: <http://www.ietf.org/rfc/rfc2893.txt> . Acesso em: 10/10/2002

[40] RFC 2461 – Neighbor Discovery for IP Version 6 (IPv6). NARTEN, T., NORDMAK, E., SIMPSON, W., December 1998. Disponível em: <http://www.ietf.org/rfc/rfc2461.txt> . Acesso em: 10/10/2002

[41] RFC 2185 – Routing Aspects of IPv6 Transition. HASKIN, D., CALLON, R..September 1997. Disponível m: <http://www.ietf.org/rfc/rfc2185.txt> . Acesso em: 10/10/2002

[42] HINDEN,R., DEERING,S., “IP Version 6 Addressing Architecture”. Disponível em: <http://www.ietf.org/internet-drafts/draft-ietf-ipngwg-temp-addr-arch-v3-11.txt>. Acesso: 17/01/2003

[43] HINDEN,R., DRAVES,R., “Default Router Preferences, More-Specific Routes and Load Sharing”.Disponível em: <http://www.ietf.org/internet-drafts/draft-ietf-ipngwg-temp-addresses-v2-00.txt>. Acesso: 17/01/2003

[44] - RFC 2401 – Security Architecture for the Internet Protocol. KENT, S., ATKINSON, R. November 19998. Disponível em: <http://www.ietf.org/rfc/rfc2401.txt> . Acesso em: 10/10/2002

[45] - RFC 2402 – IP Authentication Header. KENT, S., ATKINSON, R. November 19998. Disponível em: <http://www.ietf.org/rfc/rfc2402.txt> . Acesso em: 10/10/2002

[46] - RFC 2406 – IP Encapsulation Security Payload (ESP). KENT, S., ATKINSON, R. November 19998. Disponível em: <http://www.ietf.org/rfc/rfc/rfc2406.txt> . Acesso em: 10/10/2002

[47] – RFC 2710 – Multicast Listener Discovery (MLD) for IPv6. DEERING, S., FENNER, W., HABERMAN, B. October 1999. Disponível em: <http://www.ietf.org/rfc/rfc2710.txt> . Acesso em: 10/10/2002

[48] – Mobility Support in IPv6. Internet Draft, draft-ietf-mobileip-ipv6-08.txt. June 1999. JOHNSON, D., PERKINS, C.

- [49] – Privacy Extensions for Stateless Address Autoconfiguration in IPv6. Internet Draft, draft-ietf-ipngwg-addrconf-privacy-01.txt. October 1999. NARTEN, T.
- [50] – RFC 2462 – IPv6 Stateless Address Autoconfiguration. THOMSON, S., NARTEN, T. December 1998.
Disponível em: <http://www.ietf.org/rfc/rfc2462.txt> . Acesso em: 10/10/2002
- [51] – RFC 2526 – Reserved IPv6 Subnet Anycast Addresses. JOHNSON, D., DEERING, S. March 1999. Disponível: <http://www.ietf.org/rfc/rfc2526.txt>
Acesso em: 10/10/2002
- [52] RFC 1933 - GILLINGAN, R, NORDMARK, E. . Transition Mechanism for IPv6 Hosts and Routers. Disponível: <http://www.ietf.org/rfc/rfc1933.txt>
Acesso em: 10/10/2002
- [53] – COMER, D. Internetworking with TCP/IP - Volume I - Principles, Protocols and Architecture - 3a. edição - 1995
- [54] HINDEN, R. IP Next Generation Overview - May 14, 1995
Disponível em : <http://playground.sun.com/ipng/INET-IPng-Paper.html>
- [55] Morgan Kaufmann Publishers. Next Generation IP
Disponível em: <http://literary.com/mkp/new/3689/book/node53.html>
Acesso em: 12/12/2002
- [56] CHOWDHURY, D., Projetos Avançados de Redes IP. Rio de Janeiro: Campus – 2002.380p.
- [57] – RFC 1887 – An Architecture for IPv6 Unicast Allocation. REKHTER, Y., LI, T. December 1995.
Disponível em: <http://www.ietf.org/rfc/rfc1887.txt> . Acesso em: 10/10/2002
- [58] - RFC 2374 – An IPv6 Aggregatable Global Unicast Address Format. HINDEN, R., O'DELL, M., DEERING, S. , July 1998.
Disponível em: <http://www.ietf.org/rfc/rfc2374.txt> . Acesso em: 10/10/2002
- [59] DAVES, R., “Default Address Selection for IPv6”. Disponível em: <http://www.ietf.org/internet-drafts/draft-ietf-ipv6-default-addr-select-09.txt>. Acesso: 17/01/2003.
- [60] NORDMARK, E., DEERING, S., HABERMAN, B., “IPv6 Scoped Address Architecture”. Disponível em: <http://www.ietf.org/internet-drafts/draft-ietf-ipngwg-scoping-addr-arch-04.txt>. Acesso: 17/01/2003.
- [61] RFC 2375 – IPv6 Multicast Address Assignments. HINDEN, R., DEERING, S. July 1998. Disponível em: <http://www.ietf.org/rfc/rfc2375.txt> . Acesso em: 10/10/2002.

[62] VARELLA, W. A., "A próxima geração de protocolo de Internet", Revista FAPA, ano V, no. 6, São Paulo, Março, 2002.

[63] BIERINGER, P., Linux: IPv6 Tutorial. Disponível em: <http://www.bieringer.de/linux/IPv6>. Acesso: 14/12/2002.

[64] NETCRAFT, Site de pesquisa sobre servidores WEB", Disponível em: <http://www.netcraft.com/survey>. Acesso em: 10/04/2003.

[65] APACHE.ORG. Disponibiliza o software Apache para servidores Web. Disponível em: <http://www.apache.org>. Acesso em: 16/12/2002

[67] RED HAT. Site da distribuição Linux Red Hat. Disponível em: <http://www.redhat.com>. Acesso em: 12/12/2002

[49] NORTON, P., "Guia Completo do Linux", New Jersey, Berkerley, 2000. 567 p.