ADRIANA PAULA BORGES

Instrumentação Virtual Aplicada A Um Laboratório Com Acesso Pela Internet

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

São Paulo
2002

ADRIANA PAULA BORGES

Instrumentação Virtual Aplicada A Um Laboratório Com Acesso Pela Internet

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
2002

Borges, Adriana Paula.
Instrumentação Virtual Aplicada A Um Laboratório Com Acesso Pela Internet, São Paulo, 2002.
95 p.

Dissertação (Mestrado) – Escola Politécnica da Universidade de São Paulo. Departamento de Sistemas Eletrônicos.
1. Instrumentação Virtual 2. Laboratório Virtual 3. Monitoração Remota 1. Universidade de São Paulo. Escola Politécnica. Departamento de Sistemas Eletrônicos.

Dedico com muito carinho, mais uma etapa vencida, às pessoas que eu amo e que me fazem superar os desafios da vida com coragem e pensamento positivo: meus pais Jorge e Maria.

AGRADECIMENTOS

Agradeço a Deus por fazer da minha vida uma vitória diária, por iluminar o meu caminho, cuidar de mim, me permitir sonhar e me dar forças para buscar a realização destes sonhos.

Aos meus pais e à minha irmã Michele, meus verdadeiros super-heróis que compartilham comigo as vitórias e os maus momentos sempre com um sorriso de confiança, um abraço de coragem, palavras de incentivo.

Agradeço meu amor, Fabio Juliano, pelo carinho, companhia, compreensão e cumplicidade. Por entender os motivos que me levaram a dedicar grande parte do meu tempo exclusivamente ao trabalho.

Ao meu orientador e amigo Prof. Dr. Francisco Javier, com quem aprendi muito sobre as pessoas e o trabalho em equipe, sobre a pesquisa e a conclusão de um trabalho. Agradeço pela oportunidade, pelas discussões e incentivos, pelas repreensões tão necessárias, pela companhia e conversas nos cafés de toda tarde.

Agradeço à querida amiga Silvia, que me acompanha desde a faculdade, com quem pude contar em todos os momentos, com quem compartilhei as vitórias, as viagens, as apresentações e o dia-a-dia de pesquisadora.

Aos meus companheiros de trabalho no Grupo SIM, especialmente ao Maurício Perez, Elisabete Galeazzo, Nathália Peixoto e Henrique Peres, que com suas experiências, cada um da sua maneira, sempre me ajudaram.

Agradeço aos estagiários Rodrigo, Alexandre, Thiago, Fabio e principalmente ao Aislan Foina, Elisabeth Kinguti com quem possuo hoje uma relação de amizade que teve início por causa deste trabalho.

Agradeço à minha família e a todos que, direta ou indiretamente, contribuíram para a realização deste trabalho.

À CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior) pelo apoio financeiro.

RESUMO

Nesta dissertação são apresentados os resultados obtidos na elaboração de um sistema flexível de monitoração e controle de ensaios experimentais através de uma rede de computadores. É proposta uma infra-estrutura com base em instrumentação virtual e uma metodologia para o desenvolvimento de um laboratório virtual multi-usuários com acesso via Internet através de páginas *Web*.

São selecionadas as ferramentas que melhor se adaptam à integração entre *hardware* e *software* para possibilitar o controle de equipamentos de forma remota, através de rede de computadores utilizando um *browser Web*.

O escopo de maior abrangência deste projeto propõe a implementação de um sistema físico utilizando o acervo instrumental disponível no grupo Sensores Integráveis e Microssistemas - SIM. O principal objetivo é o controle dos parâmetros de funcionamento de dispositivos eletrônicos em um experimento genérico, através da Internet, utilizando uma interface amigável e intuitiva.

A proposta global é contribuir com a aquisição de dados a distância através da Internet para auxiliar a projetos cooperativos em áreas que incluem mas não se limitam à educação a distância, médica, industrial e comercial.

Como resultado final deste trabalho é disponibilizado um laboratório virtual que permite o acesso, o controle e a monitoração de experimentos em tempo real, em forma remota através da Internet, no endereço http://labvirtual.lme.usp.br.

ABSTRACT

This work reports results obtained in the development of a flexible system to remotely control an experiment on line through a computer network. An infrastructure based on virtual instrumentation and a methodology to develop a multi-users virtual laboratory with access through Internet through *Web* pages is proposed.

The best tools adapted to the integration between hardware and software are selected as a way to remotely control equipment through computers network using a *Web browser*.

The main purpose of this project includes the configuration of hardware, using the available instrumentation in the Integrated Sensor and Microsystems group - SIM. The main objective is to control operation parameters of electronic devices in a generic experiment, using a friendly and intuitive interface through the Internet.

The global proposal is to contribute with data acquisition through the Internet to collaborate with cooperative projects in areas that include but they are not limited to distance education, industrial and commercial environment.

As a final result, this work is available as a virtual laboratory that allows to remotely access, control and monitor experiments in real time in the Internet through the URL http://labvirtual.lme.usp.br/adriana.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

BNC – ***B****ritish* ***N****aval* ***C****onnector* ou ***B****ayonet* ***N****ut* ***C****onnector* ou ***B****ayonet* ***N****eill* ***C****oncelman*.

COM – Component Object Model.

DAQ – Data Acquisition.

DSTP – DataSocket Transfer Protocol.

FTP – File transfer protocolo.

HTML – HyperText Markup Language.

HTTP – HyperText Transfer protocol.

IP – Internet Protocol.

LAN – Local Area Network.

MAN – Metropolitan Area Network.

MIT – Massachussets Institute of Technology.

NI – National Instruments.

SCXI – Signal conditioning Multiplexer Instrument.

SDM – sistema Distribuído de controle.

TCP – Transmission Control Protocol.

UDP – User Datagram Protocol.

URL –Uniform Resource Locators, localizador uniforme de recursos.

WAN – Wide Area Network.

WWW ou WEB – World Wide Web.

1. INTRODUÇÃO

A instrumentação de um modo geral já faz parte do cotidiano das pessoas e no âmbito das atividades habituais, ela está integrada aos aparelhos eletro-eletrônicos em uma forma tal que possibilita a criação de ambientes inteligentes. O conceito de ambientes inteligentes [AL-MUHTADI, 2001] vem se consolidando tanto em ambientes industriais e comerciais como em residenciais. Entre os principais benefícios no uso de ambientes inteligentes se destacam o conforto, a segurança e a economia de uma forma geral. As formas de intervenção no ambiente normalmente é realizada através da automação por computador ou por outro tipo de controle que não exige a presença física do operador. Os aparelhos que podem ser acionados automaticamente incluem, mas não se limitam ao controle de variáveis ambientais bem como o controle de acesso e presença em sistemas de segurança.

Os eletrodomésticos como TV, geladeira e o forno de microondas, estão no alvo dos principais fabricantes de microchips (*Sun Microsystems*, por exemplo) para computadores. A razão do interesse é a possibilidade de implementar uma rede doméstica onde todos os aparelhos podem estar ligados entre si e conectados à Internet. Já é realidade no *Massachussets Institute of Technology* (MIT) [LINK 1], onde pesquisadores estudam seriamente recursos móveis e trajáveis de computação. Por meio de uma malha especial e um conjunto de sensores espalhados pelo tecido, o sinal é enviado para o computador central de uma casa que terá o ar-condicionado regulado a partir da temperatura de quem está vestindo a jaqueta. É possível, através desta tecnologia, operar a luminosidade dos ambientes, checar as condições dos alimentos guardados na geladeira e acionar sistemas de segurança, como alarmes e travas elétricas de portas e cofres.

Trabalhos semelhantes, supérfluos aos olhos daqueles que resistem ao advento das máquinas, são tidos como pesquisas fundamentais pela indústria, que busca incessantemente novos padrões para determinar o rumo da nossa sociedade de consumo nos próximos anos. Embora os computadores ainda pareçam tecnologicamente

dimensionados em excesso e pouco úteis para as atividades habituais do cidadão comum, outros recursos provenientes das tecnologias de transmissão via Internet, com ou sem fio, já fazem parte do cotidiano das pessoas.

A instrumentação virtual foi originalmente concebida como uma ferramenta de desenvolvimento que automatiza procedimentos laboratoriais e industriais. Os métodos convencionais de instrumentação, além de onerar os orçamentos, não satisfaziam as necessidades de se ter diversos controles de dados e sinais ao mesmo tempo [TANER, 1997]. As restrições da instrumentação convencional, junto às econômicas, permitiram que a instrumentação virtual estendesse seu raio de ação e fosse inserida em computadores pessoais através de placas padronizadas para a aquisição de dados [GOLDBERG, 2000]. Desta forma, a instrumentação virtual mostra-se como uma ferramenta que oferece flexibilidade, com ampliados horizontes de aplicação.

O uso da instrumentação virtual tem se destacado também no contexto de controle e monitoração de processos no âmbito industrial, na pesquisa e desenvolvimento da eletrônica embarcada [BECK, 2001]. A popularização da Internet com sua facilidade de uso e possibilidade de integração surge como a principal ferramenta de apoio para o desenvolvimento de sistemas de controle que podem ser monitorados a distância.

A Internet tem se tornado popular em todos os sentidos. Essa rápida expansão se deve: à facilidade de comunicação e de busca de informações em diversos lugares do mundo; facilidade de conexão e possibilidade de acesso a grandes bases de dados; além da possibilidade de implementar e gerenciar aplicativos com interface homem-máquina interativa e intuitiva, que está sendo amplamente explorada e utilizada para aplicações nas mais diferentes áreas do conhecimento.

A consolidação do uso da Internet como ferramenta de compartilhamento de informações, juntamente com o desenvolvimento de ferramentas modernas de controle, aquisição e distribuição de dados via rede de computadores, tornam possível e vêem impulsionar o controle e acompanhamento de ensaios em ambientes experimentais remotos [BROFFERIO, 1998].

Além dos avanços da Internet é possível acompanhar também a consolidação do uso de microcomputadores para os mais diversos fins, envolvendo desde simples tarefas de escritório até complicados controles de processos industriais. Isso se deve ao emprego das redes de computadores e das novas tecnologias que surgem a cada dia na tentativa de automatizar processos simples ou complexos.

O desenvolvimento e uso de laboratórios virtuais onde estudantes e pesquisadores controlam instrumentos, muitas vezes específicos, de forma remota onde existe uma metodologia de coleta e análise de dados se mostra como uma das recentes atividades promissoras que une diferentes tecnologias para a elaboração de sistemas de controle sofisticados oferecendo certa segurança na troca de informações. Já existem, em diversas partes do mundo, laboratórios virtuais que fazem uso da Internet para compartilhar informações entre equipes de trabalho na área industrial, entre alunos e pesquisadores na área acadêmica [LINK 2]-[LINK 6], e entre equipes médicas [REGGIANI, 2000]. Esses laboratórios são implementados em diversas áreas do conhecimento e com os mais diferentes propósitos. Podem existir apenas para consultas a bancos de dados bem como para simular ou controlar experiências e monitorar processos de controle de forma remota. Porém, os pontos de interesse deste trabalho são os laboratórios virtuais onde o controle da instrumentação é realizado via Internet.

Outros pontos importantes que incentivam a realização deste trabalho são: a possibilidade de disponibilizar resultados de exames médicos para serem analisados por vários especialistas como, por exemplo, a monitoração cardíaca [MAGRABI, 1999]; e principalmente a disponibilização de experimentos laboratoriais para alunos geograficamente distantes do ambiente escolar.

Como já antes mencionado, existe uma tendência significativa do uso da Internet como o veículo de comunicação interativa no meio acadêmico bem como nas indústrias na atualidade. Baseando-se nesta tendência foi idealizado este projeto que faz uso da Internet, aproveitando seu potencial para compartilhar instrumentação, de forma a torná-la mais flexível e disponível aos usuários da grande rede de computadores.

A troca de informações médicas sobre pacientes entre hospitais e grupos de especialistas para avaliação, discussão e diagnóstico de doença é hoje uma realidade e

conquista, a cada dia, maior atenção dos especialistas e analistas de sistemas. Como exemplo pode ser citado o InCor (Instituto do Coração do hospital das Clínicas de São Paulo) que utiliza um sistema de monitoramento de sinais vitais em tempo real de forma remota para pacientes recentemente operados [REGGIANE, 2000].

O conceito explorado neste trabalho, instrumentação controlada via Internet, aplica–se ao desenvolvimento de sistemas inteligentes envolvendo redes de computadores voltadas para automação predial e/ou residencial, educação a distância onde existem atividades laboratoriais, na eletrônica embarcada, possibilitando o controle e calibração de dispositivos sensores de forma remota, entre outros.

O uso de laboratório virtual para aulas práticas, onde ocorre a manipulação de instrumentos de medida e aquisição de dados em tempo real, é uma forma didática recente e aplicada em universidades distribuídas pelo mundo, como por exemplo, Universidade Nacional de Singapura (NUS) [HOON, 1998], Universidade de Stanford [LINK 5], Universidade de Padova [BERTOCCO, 1998], Universidade do Estado de Portland [STEGAWSKI, 1998], Universidade Politécnica de Valência [PALOP, 2000], Universidade Técnica de Milano [BROFFERIO, 1998]. Torna-se uma maneira de estimular o uso da Internet para absorver conhecimento, e uma forma de implementar aplicativos para áreas carentes deste tipo de sistema de controle e integração de grupos de trabalho conjunto.

A proposta deste trabalho é explorar a execução testes de laboratórios via *browser Web* permitindo que os usuários controlem o aparato experimental, ou seja, os equipamentos fisicamente localizados no laboratório real, via Internet obtendo informações em "tempo real" através de uma interface homem-máquina interativa [BORGES, 2000 a]. O objetivo principal deste projeto é estabelecer a conexão entre equipamentos e microcomputadores em rede local e remota via instrumentação virtual, para permitir o acesso interativo aos controles e o acompanhamento de experiências. O objetivo final e de maior abrangência deste trabalho é a disponibilização de ensaios experimentais em modo totalmente interativo para usuários conectados à Internet.

A possibilidade de disponibilizar equipamentos, muitas vezes de alto custo, e compartilhar projetos entre grupos de pesquisa de outras localidades através da Internet são as principais motivações para este trabalho.

No segundo capítulo são abordados alguns dos Laboratórios Virtuais pesquisados e ativos na atualidade que se mostram como os mais significativos para este trabalho. São abordados também aspectos da instrumentação virtual, a utilização das redes de computadores como ferramenta para implementar sistemas distribuídos de controle através da Internet que estão sendo abordados pela literatura.

No terceiro capítulo são apresentados os materiais utilizados para tornar a proposta deste trabalho um sistema real, que permita o controle de ensaios experimentais a distância. Neste capítulo é apresentada a metodologia utilizada na implementação do sistema que torna possível o controle de instrumentos em tempo real através da rede *WWW*.

A descrição detalhada do sistema desenvolvido bem como os resultados experimentais obtidos junto com o aparato experimental implementado são apresentados no quarto capítulo.

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

A bibliografia é dividida entre publicações como artigos e livros além das páginas da Internet (*Sites*).

2. LABORATÓRIO VIRTUAL

Este capítulo apresenta conceitos básicos sobre sistemas que permitem o controle de instrumentação via Internet além da descrição de alguns dos laboratórios virtuais pesquisados que serviram de base para o desenvolvimento deste trabalho. São apresentadas também as tecnologias utilizadas no desenvolvimento de laboratórios virtuais acessíveis através de páginas *Web*.

2.1 Instrumentação Virtual

O crescente aumento na utilização da instrumentação tem provocado mudanças nos antigos procedimentos de controle e medidas onde o usuário deveria ajustar os parâmetros dos instrumentos físicos clássicos, que executavam tarefas específicas, para que testes fossem realizados. Havia a necessidade de melhorar a utilização oferecendo maior liberdade de ação de forma a tornar os instrumentos físicos transportáveis, ou seja, uma forma de instrumentação mais flexível. As restrições da instrumentação convencional de concepção monolítica, junto às econômicas, permitiram que a instrumentação virtual estendesse seu raio de ação e fosse inserida em computadores pessoais através de placas padronizadas para a aquisição de dados [TAN, 2001], como mostrado na **Figura 1** onde é possível observar um microcomputador, uma placa de aquisição de dados, um módulo BNC, um adaptador, um registrador e os sensores que geram sinal.

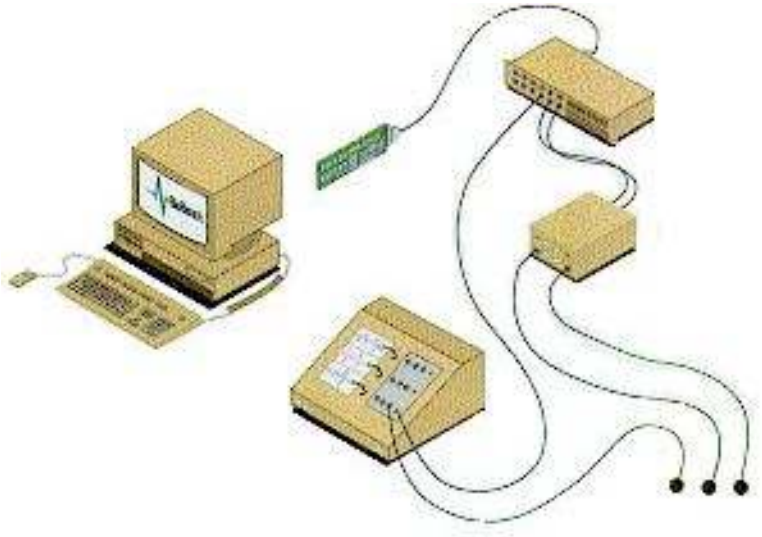

Figura 1 - Exemplo genérico de ferramenta de instrumentação inteligente onde observamos o uso de um *software* em um computador controlando equipamentos de medida através de uma placa de aquisição de dados.[1]

Em princípio a instrumentação virtual, cujo conceito está representado na **Figura 2**, é uma proposta para controle de processos, envolvendo equipamentos de medida, *software* e computador, tornando possível a realização de controle do fluxo de informações entre dispositivos através de interfaces padronizadas para comunicação, entre homem-máquina e máquina-máquina. O nome "instrumento virtual" deriva dos aspectos realistas de operação que dizem respeito aos instrumentos clássicos [CRISTALDI, 1999] como, por exemplo, voltímetros, osciloscópios entre outros, e foi originalmente concebida como uma ferramenta de desenvolvimento que automatiza procedimentos laboratoriais e industriais [MARINO, 2000].

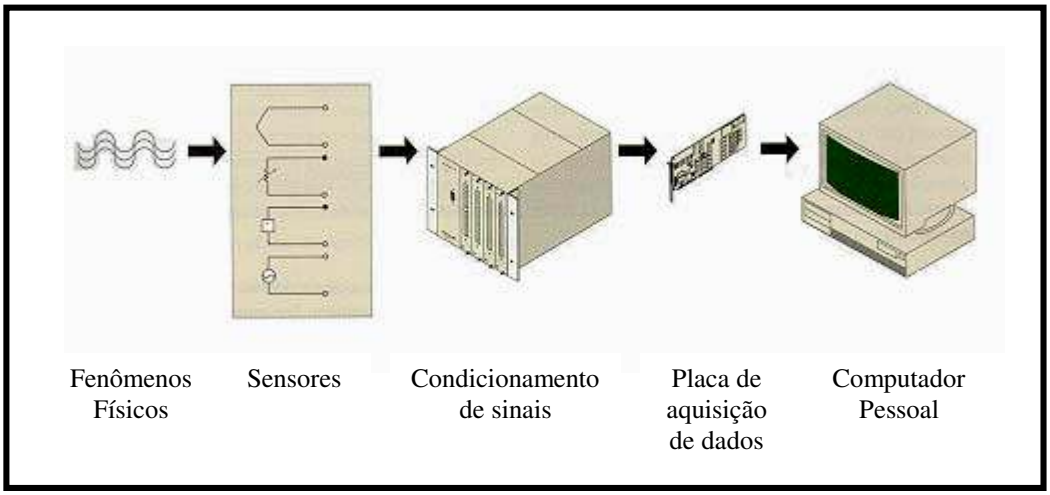

Figura 2 - Exemplo genérico de ferramentas de instrumentação virtual (placas de aquisição, computadores, software), representando a instrumentação virtual.2

2.1.1 Parâmetros que descrevem a instrumentação

O sistema de instrumentação virtual pode ser descrito por quatro parâmetros: modelo de programação, estrutura de controle, latência e dispositivos específicos [SPOELDER, 1999]. O primeiro parâmetro está relacionado ao modelo de programação – a programação é gráfica, ou seja, não é necessário digitar linhas de códigos para a execução de determinada atividade. O segundo parâmetro que descreve a instrumentação virtual é a estrutura de controle – diagramas de blocos que executam seqüências lógicas são dispostos de forma a executar o controle de um processo como é possível verificar na **Figura 3**. A latência, citada como terceiro parâmetro, está relacionada ao tempo de resposta entre a mensagem do *software* e os instrumentos, ou seja, o tempo em que um instrumento leva para responder a um comando vindo de um *software*. E para completar, os dispositivos específicos – como, por exemplo, osciloscópio, fonte de tensão formam o conjunto descrito como o quarto parâmetro de um sistema instrumentação virtual.

[1] Figura retirada do catálogo da *National Instruments* 1999 – página 43.
[2] Figura retirada do catálogo da *National Instruments* 1999 – página 197.

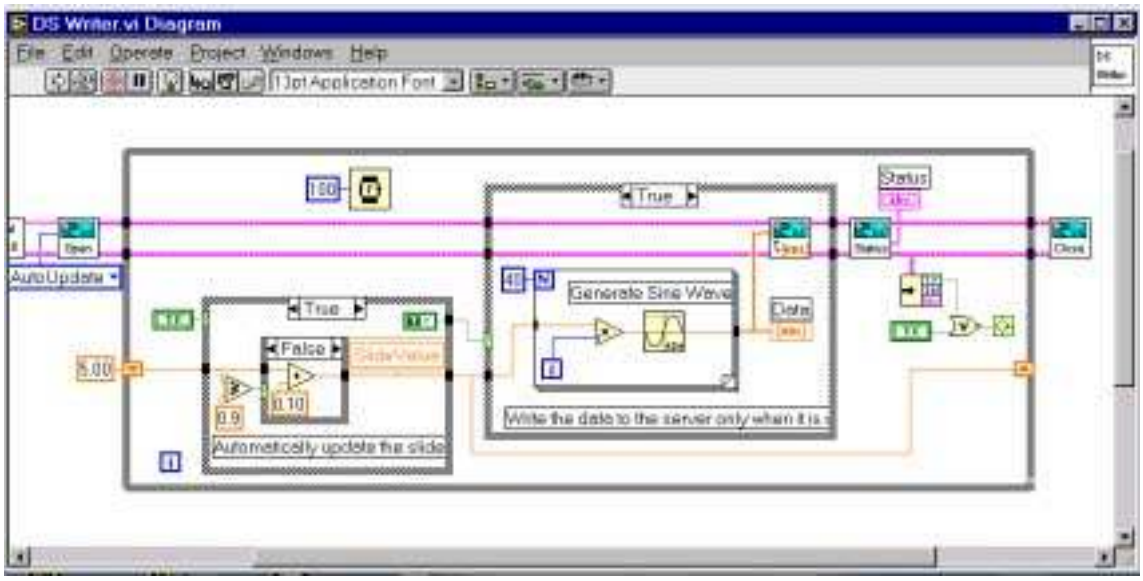

Figura 3 - Exemplo de diagrama de blocos de um instrumento virtual genérico (VI – virtual instrument) que acompanha a Ajuda do *software LabVIEW®* versão 5.1, aqui demonstrado para ilustrar a utilização do diagrama.

2.1.2 Hardware e Software

Os microprocessadores e técnicas de *software* têm tornado a instrumentação virtual uma forma sofisticada de controle, e muitos padrões de comunicação entre *hardware* têm sido estabelecidos [WANG, 2000], [TANER, 1997] para oferecer suporte aos dispositivos e interface para instrumentação.

A eficiência do projeto da instrumentação virtual está relacionada ao próprio processo de programação, descrito anteriormente, onde o usuário não precisa ser especialista para implementar o controle de um instrumento. Para que alguma atividade seja executada, basta conectar dois componentes gráficos dispostos em um painel frontal, exemplificados na **Figura 4**. As interfaces assemelham-se aos *displays* dos instrumentos reais, ou seja, fazem uso da instrumentação virtual para que os indivíduos acostumados a trabalhar com instrumentação convencional também possam se adaptar a um novo tipo de ambiente [CRISTALDI, 1999] onde é possível realizar controles e

medições de diversos parâmetros simultaneamente, o que propicia a transportabilidade da instrumentação convencional.

O *software* que representa a utilização da instrumentação virtual e que se mostra como importante ferramenta para o desenvolvimento deste projeto é o *LabVIEW®* [LINK 7], ambiente de programação gráfica que combina programação flexÍvel, possibilitando a integração entre hardware e software.

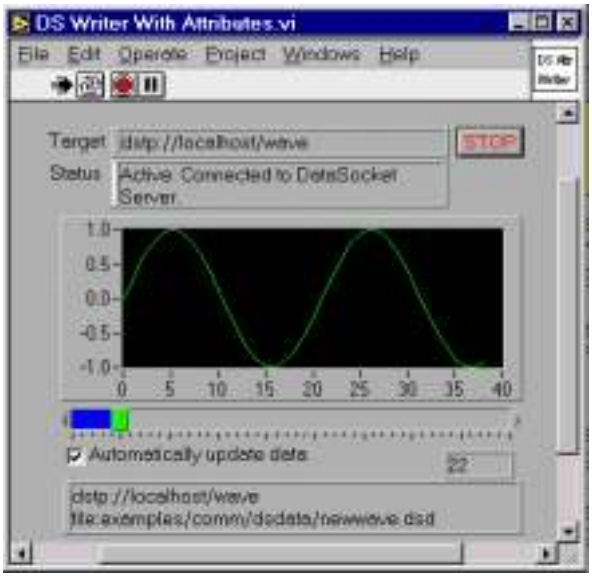

Figura 4 - Exemplo de painel frontal de um instrumento virtual (VI – virtual instrument) que acompanha a Ajuda do *software LabVIEW®* versão 5.1.

Na programação gráfica, o diagrama de blocos demonstra a lógica estabelecida para resolver um determinado problema através da conexão de ícones para implementar um determinado controle. Esta estrutura reflete-se também na manutenção e documentação de todo programa gerado, reduzindo, desta forma, os erros de programação e interpretação dos mesmos.

Wang [WANG, 2000] afirma que as interfaces elaboradas para controle via instrumentação virtual, são responsáveis por:

- Manter contato com a unidade central de processamento e executar diferentes testes e funções para medição de sinais. Executar o controle de

fluxo de informação de dispositivos periféricos (como sensor e circuitos de processamento de sinal) através de interfaces padrões.

- Adquirir mais de um tipo de dado e mostrar o resultado desta aquisição; acompanhar o fluxo de informação do controle de dispositivos periféricos (como sensor e circuitos de processamento de sinal) de forma a permitir futuras análises. Possibilitar a exibição e análise de dados adquiridos.
- Gerenciar e arquivar dados, a impressão, a comunicação dos equipamentos em rede, o acesso à Internet, ou seja, gerenciar dados de forma a permitir a troca de mensagem em rede local e/ou remota.

2.1.3 Motivação

Dentre as vantagens que a instrumentação virtual oferece destacam-se a flexibilidade e a redução de tempo necessário para implementação dos controles. Representando uma nova geração, os instrumentos virtuais estão sendo incorporados aceleradamente ao desenvolvimento de aplicações na indústria em geral.

Uma outra vantagem relaciona-se ao fato da instrumentação estar baseada em um computador genérico possuindo, desta forma, a capacidade de processamento de dados limitada ao *hardware* do computador em termos de velocidade de processamento da informação e configuração de entrada e saída. Caso o sistema opere em um ambiente de rede, a taxa de transmissão de dados torna-se dependente da arquitetura da rede [WANG, 2000]. Este fato envolve um estudo sobre a rede de computadores onde o sistema irá ser executado. Os computadores com sistemas multi-processadores possibilitam o uso de sistemas de medida complexos superando a limitação de um único computador. Esta limitação deve-se ao poder de computação global ou ao número de sinais adquirido e processado. Os sistemas de processamentos distribuídos baseados em redes locais podem ser adotados para monitorar e controlar aplicações quando há a necessidade de processar mais de dois sinais ou quando os pontos de aquisição estão fisicamente distantes, o que traz uma redução no custo de interconexão entre o sistema de processamento e os sensores distantes.

2.1.4 Sistemas Distribuídos de Medição

A instrumentação apresenta-se como uma ferramenta sofisticada para aquisição e monitoração que pode ser aplicada em diferentes áreas do conhecimento, mostrando-se como base eficiente e intuitiva na comunicação homem-máquina envolvendo redes de computadores.

Existe uma crescente necessidade das indústrias e de laboratórios de pesquisa na implementação de sistemas de medida com flexibilidade, adaptabilidade e capacidade de programação para automatizar os procedimentos de medida em aplicações complexas que podem ser distribuídas [CRISTALDI, 1999], ou seja, executadas através de redes de computadores com diferentes pontos de aquisição e gerenciamento da informação.

Do ponto de vista de sistema ideal, ou seja, um sistema que ofereça ampla flexibilidade de acesso concatenando as tecnologias de instrumentação, rede e Internet, pode-se imaginar a implementação de um sistema distribuído de medição - SDM. Um SDM é composto por dispositivos computacionais heterogêneos, redes de comunicação, e um sistema de controle de instrumentos [KATCHABAW, 1999] com aplicações distintas. Porém os sistemas de controle distribuídos apresentam limitações consideráveis que restringem sua aplicação em laboratórios virtuais, sendo a primeira relacionada ao número de centrais de processamento distribuídas na rede, seja esta de pequena ou grande proporção. O uso de subsistema de aquisição em rede permite ampliar a capacidade de aquisição da unidade de processo, porém não devem ser esquecidas as interferências nos sinais que trafegam por uma rede de computadores [CRISTALDI, 1999]. Pode haver ruído cujo grau de interferência está relacionado ao meio físico de interligação dos computadores.

Uma segunda restrição dos SDM s está relacionada à área operacional coberta por uma única unidade de processamento [LEE, 1999]. Não é recomendável que os sensores e subsistema de aquisição estejam geograficamente distantes da fonte geradora de sinais elétricos, de acordo com a especificação de qualidade oferecida pelas placas de aquisição.

As limitações dos sistemas distribuídos de medição estão relacionadas também ao desempenho das redes de computadores e todos os problemas que estas apresentam, como por exemplo, atrasos devido ao intenso tráfego de informações em determinados horários do dia. O tópico seguinte oferece uma breve conceituação de redes de computadores e o ganho na transmissão de informação oferecido por cada tipo de rede.

2.2 Redes de Computadores

O crescimento do uso de redes de comunicação tem permitido a troca de informação de forma rápida e segura. A comunicação de dados tornou-se parte fundamental da computação. As redes de computadores transportam dados sobre assuntos diversificados e são essenciais porque permitem que pessoas compartilhem experiências enviando programas, mensagens ou arquivos, disseminando informação em todas as áreas do conhecimento. Este item tem a finalidade de expor alguns conceitos que serão citados continuamente no decorrer do trabalho bem como familiarizar o leitor com os termos relativos ao sistema elaborado e descrito no capítulo 4 neste trabalho.

Nos últimos anos foi desenvolvida uma tecnologia para possibilitar a interconexão de redes físicas diferentes que operem como uma unidade coordenada. Essa tecnologia, chamada também de arquitetura de redes, acomoda distintas tecnologias básicas de *software* e *hardware*, proporcionando uma forma de interconectar computadores e um conjunto de convenções que possibilitam as comunicações entre máquinas diferentes [ORFALL, 1996].

A tecnologia de redes está relacionada à interligação física de computadores. Essas redes podem ocupar diferentes extensões e interligar máquinas geograficamente próximas ou distantes. Para caracterizar as diferenças entre extensão e capacidade dos tipos de redes, estas foram divididas em três categorias: redes de pequena distância ou redes locais, as *LANs* (L*ocal Area Network*); as redes de extensão intermediária conhecida como *MANs* (*Metropolitan Area Network*); e as redes de longa distância também chamadas de *WANs* (*Wide Area Network*) [STALLINGS, 1997]. Os tipos que

merecem destaque e servem como material para este trabalho são a rede local e a de longa distância

A rede local, que interliga salas em um edifício comercial ou prédios de um campus universitário, é o tipo mais comum de rede de computadores. Até mesmo quem tem dois computadores ligados em sua própria casa possui uma rede local. Com a expansão das redes foi viabilizada a interconexão de diferentes redes locais, dando origem às redes metropolitanas e redes remotas. As redes locais caracterizam-se por altas taxas de transferência (que pode variar entre 10 Mbps - Megabits por segundo - e 2 Gbps – gigabits por segundo), baixo índice de erros e custo relativamente pequeno devido à extensão (1 km) [TANENBAUM, 1987].

O conceito de rede metropolitana pode parecer um tanto quanto confuso no que diz respeito às diferenças existentes entre uma *MAN* e uma rede remota. Na verdade, a definição para este tipo de rede de computadores surgiu depois das *LAN*s e *WAN*s. Ficou estabelecido que redes metropolitanas são aquelas que estão compreendidas numa área metropolitana com extensão de 10 km, como as diferentes regiões de toda uma cidade [TANENBAUM, 1987]. Normalmente redes metropolitanas são constituídas de equipamentos sofisticados, com um custo alto para a sua implementação e manutenção, que compõem a infra-estrutura necessária para o tráfego de som, vídeo e gráficos de alta resolução. Por serem comuns nos grandes centros urbanos e econômicos, as rede metropolitanas são o primeiro passo para o desenvolvimento de redes remotas.

Redes remotas são aquelas que cobrem regiões extensas. Na verdade redes remotas são um agrupamento de várias redes locais e/ou metropolitanas, interligando estados, países ou continentes. Tecnologias que envolvem custos elevados são necessárias, tais como cabeamento submarino, transmissão por satélite ou sistemas terrestres de microondas. As linhas telefônicas, uma tecnologia que não é tão sofisticada e nem possui um custo muito elevado, também são amplamente empregadas no tráfego de informações em redes remotas. Este tipo de rede caracteriza-se por apresentar uma maior incidência de erros, e também são lentas. A velocidade típica de transmissão de informações em uma rede *WAN* é influenciada principalmente pelo número de máquinas utilizando a rede simultaneamente [TANENBAUM, 1987].

Novas técnicas estão surgindo de modo a reverter esses problemas, mas a sua implementação depende de toda uma série de fatores, logo o processo é gradativo. Um exemplo de rede remota muito popular é a Internet, que possibilita a comunicação entre pessoas de lugares totalmente diferentes.

O grande avanço tecnológico atual, as redes de computadores e em especial a Internet, que permite conectar pessoas distribuídas por todo o mundo, têm sido a motivação para o uso da tecnologia de computadores em diversas áreas do conhecimento, facilitando a interação e a troca de informações entre as pessoas.

A computação sofisticada, ou seja, computadores com maior velocidade de processamento e mais memória *RAM* (*Random Access Memory* – memória de acesso randômico - cujo tamanho, 64 ou 128 MB – *MegaBytes*, varia dependendo da configuração de cada microcomputador), e o aumento da largura de banda na comunicação em rede proporcionam o desenvolvimento de conteúdos com melhor qualidade e aplicativos mais complexos [SPOELDER, 1999] para funcionamento e acesso remoto.

Uma forma de automatizar ainda mais os processos de aquisição de dados é poder fazê-los através da Internet, o que facilita o controle, além de ser uma forma que oferece segurança, quando tratamos de ambientes insalubres, e flexibilidade comprovados por diversos sistemas existentes hoje [PATON, 1998].

A Internet, pela facilidade de uso e conectividade que oferece, se mostra como um meio ideal para a experimentação remota [MALY, 1998].

A utilização de rede de subsistema de aquisição e instrumentos pode ser útil para superar a degradação induzida por ruído ambiental nas conexões diretas entre sensor e as placas de aquisição instaladas na unidade de processamento.

Para escolher a rede a ser utilizada, é necessário levar em conta fatores como localização dos dispositivos de controle e unidade de processamento, meio físico por onde o sinal vai trafegar em rede, e o protocolo de comunicação empregado nesta comunicação. Um estudo sobre o meio físico, sobre os cabos de interligação da rede deve ser feito para identificar qual tipo oferece maior ou menor ruído. Isso envolve a análise custo X benefício, a possibilidade de implantação de uma arquitetura de rede

eficiente e utilização de um protocolo que auxilie no cumprimento dos objetivos estabelecidos relacionados à transferência da informação.

2.3 Internet e o serviço *World Wide Web*

A Internet é uma rede de computadores que surgiu em 1969, de um projeto do Departamento de Defesa (*DoD - Department of Defense*) dos Estados Unidos, inicialmente denominada *ARPAnet* (*ARPA - Advanced Research Projects Agency*), cujo objetivo era possibilitar a interligação de computadores distribuídos em pontos estratégicos, utilizados em centros de investigação com fins militares [HAHN, 1995].

A Internet consolidou-se como veículo de comunicação nesta década. A rede tornou-se um importante meio com capacidade para difusão instantânea de informação cuja característica fundamental é a capacidade de funcionar alternativamente por diferentes canais de comunicação.

2.3.1 Como funciona a Internet

Na Internet estão interligados vários tipos de redes, quer quanto aos equipamentos que as compõem e à tecnologia utilizada, bem como à cobertura geográfica por ela ocupada (redes de longa distância, redes locais já abordadas neste trabalho). Para que a comunicação seja possível é necessária a utilização de protocolos. Um protocolo é um conjunto de regras pré-estabelecidas que possibilitam a comunicação entre computadores distantes rodando sistemas diferentes [HAHN, 1995]. O conjunto de protocolos (também designado família de protocolos) utilizados na Internet é o *TCP/IP* (*Transmission Control Protocol/Internet Protocol*) [COMER, 1988].

Para que a comunicação entre os diversos computadores interligados na Internet não chegue ao destino errado, ou não seja perdido pelo caminho, é atribuído um endereço numérico a cada computador conectado à rede, também chamado de *host,* que

o identifica univocamente. Esse endereço, denominado endereço IP, é constituído por quatro campos, separados entre si por pontos, como por exemplo: 192.168.160.200.

2.3.2 Os serviços Internet

Os serviços disponibilizados na Internet são serviços distribuídos, baseados no modelo cliente/servidor. Isto é, existem pelo menos duas peças de *software* que interagem para prestar um determinado serviço.

Uma maneira simples de verificar o funcionamento do modelo cliente/servidor [VAUGHN, 1994] é a forma como é possível buscar uma informação: o *browser* de *WWW* (o cliente) ou navegador da Internet solicita ao computador da rede onde está armazenada a página que pretende-se visitar (o servidor, onde está instalado o software servidor de *WWW*). O servidor, por sua vez, recebe a solicitação procura a informação armazenada e a envia ao *host* solicitante através do número *IP*. Ao receber a informação do servidor, o cliente de *WWW* interpreta-a para exibi-la com a formatação correta.

2.3.2.1 A *World Wide Web*

A *WWW* é o ambiente hipertextual no qual toda a informação existente na Internet, sejam textos, imagens, serviços computacionais, etc., podem ser acessada de forma simples e consistente, usando sempre a mesma ferramenta - o seu cliente/*browser* de *WWW*. A *WWW* é um serviço distribuído, baseado no modelo cliente/servidor, tal como a maioria dos serviços da Internet.

Na *Web*, como também é conhecida a *WWW*, é possível acessar servidores em qualquer ponto do mundo através de apontadores (*links*) de hipertexto, utilizar diferentes sistemas e acessar diferentes tipos de informação e serviços da Internet, sem a necessidade de mudar de *software* (programa navegador) sempre que muda de serviço.

A técnica de hipertexto possibilita a interligação e a navegação entre diferentes recursos e serviços. A *World Wide Web* permitem-nos acessar praticamente todos os

recursos existentes na Internet, podendo constituir uma espécie de interface universal para a rede.

Inicialmente a troca de informações na rede era normalmente feita por correio eletrônico ou por protocolos de transferência de arquivos (*File Transfer Protocol* - o *FTP*).

Em 1992, três anos após o início do desenvolvimento do conceito de *World Wide Web* pelos cientistas do laboratório nuclear suíço *CERN* (*Conseil Europeen pour la Recherche Nucleaire* – antiga denominação do atual Laboratório Europeu de Partículas Físicas) [LINK 8], o Centro Nacional para Aplicações em Supercomputadores (*NCSA – National Center for Supercomputing Applications*), sediado em Chicago, deu início a um projeto visando a criação de uma interface amigável para a comunicação via Internet. O *Mosaic*, primeiro *browser* em modo texto para acesso a bases de dados de hipertexto, foi desenvolvido por Marc Andreessen [LINK 9].

Graças às inovações introduzidas pelo *Mosaic*, o usuário passou a contar com um programa (*software*) de visualização que permite a apresentação de textos, imagens e gráficos de uma forma atraente como a de uma página de revista.

2.3.2.2 Como funciona a *World Wide Web*

A *World Wide Web* baseia-se num conjunto de princípios básicos, respeitados de comum acordo, sem necessidade de autoridade central, onde um dos pontos importantes é a utilização dos mesmos protocolos e mecanismos pelos servidores e clientes, entre os quais se destacam:

HTTP - Hypertext Transfer Protocol - protocolo usado como mecanismo de transporte na *WWW*. Suporta a transferência de todos os tipos de informação multimídia.

URLs - Uniform Resource Locators - mecanismo que através de um endereço nos dá acesso a recursos na Internet, é o que nos permite o acesso a outros tipos de recursos/serviços tais como, *FTP* (*File Transfer Protocol* – protocolo de transferência de arquivo, News, etc., no espaço *WWW*.

HTML - HyperText Markup Language **-** Linguagem de formatação para hipertexto usada pelos clientes de *WWW*, na qual são construídos os documentos para a *Web*.

O princípio de funcionamento da *WWW* pode ser divido nas etapas identificadas na **Figura 5**:

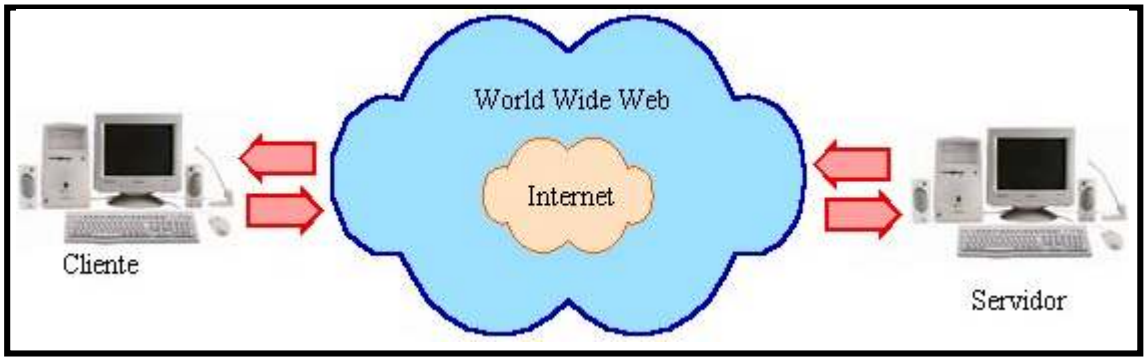

Figura 5 - Princípio de funcionamento da *World Wide Web* – cliente/servidor.

O início do funcionamento da *Web* ocorre com a solicitação de uma informação. Um pedido é interpretado pelo *browser* como sendo o objeto conhecido pelo *URL*. O *browser* envia o pedido para o servidor *Web* – máquina ligada à Internet onde estão armazenados os dados, as páginas com informações para que possam ser acessadas. O servidor *WWW*, ou servidor *Web* localiza o objeto solicitado e o envia para o cliente. Como este documento inclui diversos objetos, como imagens, por exemplo, o servidor envia cada um desses objetos individualmente. O *browser* recolhe os diferentes objetos do documento solicitado, os interpreta e apresenta a informação organizada em um único objeto formando finalmente a página solicitada.

2.4 Sistemas Cliente/Servidor

A arquitetura de sistemas de controle, em laboratórios virtuais, encontrada na literatura é baseada na comunicação cliente/servidor, conforme ilustrado na **Figura 6**. A arquitetura cliente/servidor é composta por dois ou mais computadores em um sistema no qual o processamento da informação é dividido em módulos ou processos distintos de modo a executar dois serviços: Servidor – cuja função é disponibilizar serviços aos usuários do sistema; e Cliente – cuja função é permitir aos usuários o acesso aos serviços disponibilizados pelo servidor. Sendo assim um processo é responsável pela manutenção da informação (servidores) e outros responsáveis pela obtenção dos dados (os clientes).

Os processos cliente enviam pedidos para o processo servidor, e este por sua vez processa e envia os resultados dos pedidos aos clientes solicitantes. Assim, em uma rede cliente – servidor, pode existir somente uma estação servidora de disco, uma servidora de impressora, uma servidora de correio eletrônico, etc.

Dentre as vantagens que podem ser destacadas na utilização da arquitetura cliente/servidor estão:

1. **Escalabilidade** - Um sistema cliente/servidor pode ser expandido verticalmente pela adição de mais recursos adicionados ao computador, com a função de disponibilizar informação para qualquer cliente solicitante.
2. **Independência de plataformas -** Os sistemas cliente/servidor não ficam presos a um ambiente de *software* ou *hardware*.
3. **Flexibilidade -** Com a possibilidade de vários clientes, em diferentes pontos da rede, estarem acessando informações em um único servidor.
4. **Fácil Acesso aos Dados -** Como é o processo cliente que gerencia a interface, deixando o servidor livre para manipular os dados, este por sua vez fica mais disponível.
5. **Redução de Custos Operacionais -** Como os custos de *hardware* e *software* estão constantemente sendo reduzidos, a troca dos sistemas grandes por sistemas com redes integradas pode ser feita com um baixo custo.

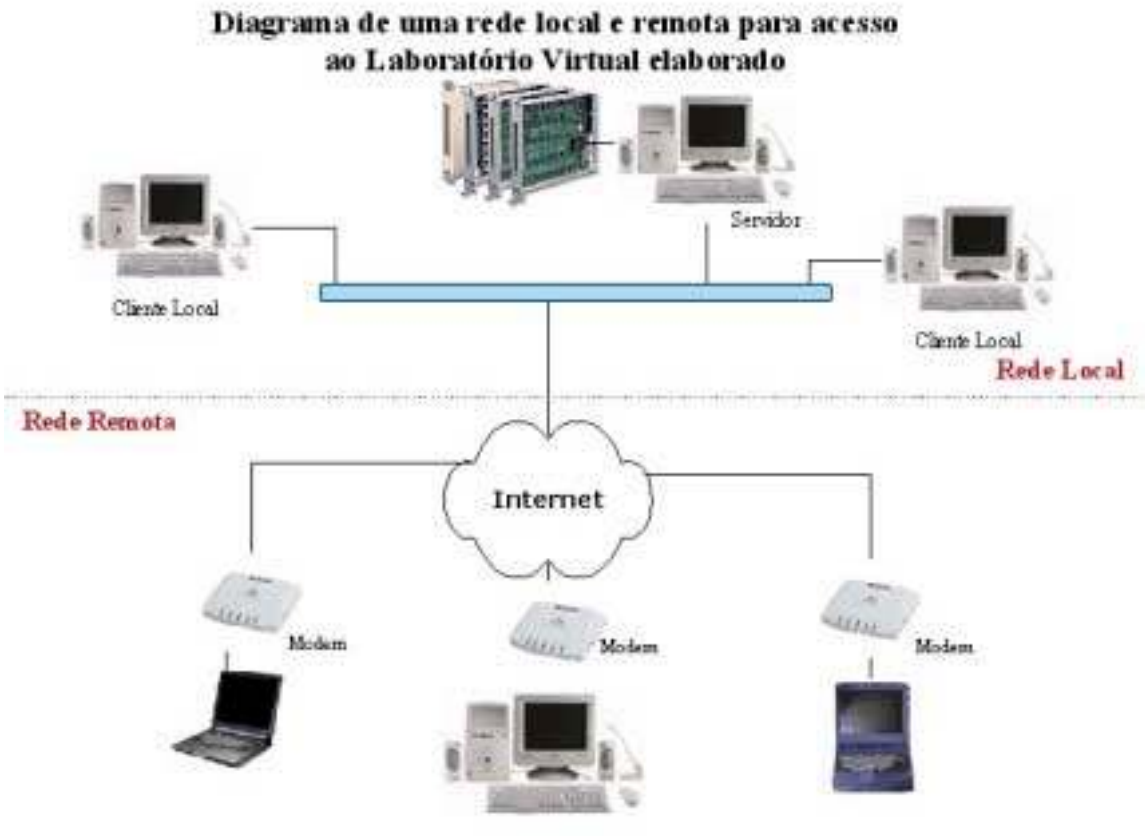

Figura 6 - Esquema de arquitetura de rede cliente/servidor para controle de instrumentos via rede de computadores.

A implementação de sistemas de controle para Internet faz uso da arquitetura cliente/servidor. O servidor, situado próximo ao experimento, transfere comandos transmitidos pelo cliente [OVERSTREET, 1999].

A estação servidora normalmente está dedicada a essa função, e sua utilização por usuários é desaconselhada por sobrecarregar o sistema operacional. Os motivos básicos que levam a isso são a segurança, pois evita que, caso um usuário esteja utilizando a estação servidora e execute alguma aplicação que ocupe a máquina desnecessariamente, comprometendo desta forma o desempenho da rede que pode ser alterado quando diferentes tipos de aplicativos são executados ao mesmo tempo (dependendo do sistema operacional), prejudicando o funcionamento da rede como um todo.

No caso dos laboratórios virtuais pesquisados, o servidor é o responsável pelo controle dos instrumentos, recebendo e enviando comandos vindos do cliente para alterar, por exemplo, o estado dos dispositivos de medida. Desta forma, a comunicação entre os computadores se traduz em um esquema de interação entre os processos cliente e servidor resultando em uma troca cooperativa [BERTOCCO, 1998]. Com esta técnica os usuários podem compartilhar informações de experimentos remotos interagindo com os programas controlados pelo servidor através da rede.

2.5 Laboratórios Virtuais

O uso da Internet e da tecnologia *World Wide Web* em laboratório de medidas interativo é desenvolvido com o propósito de fornecer flexibilidade no controle de processos industriais e na educação de alunos em localidades distantes da sala de aula ou da universidade [ALTPETER, 1995]. A tecnologia *World Wide Web*, baseada nos princípios de hipertexto está altamente focada na interface homem-máquina, oferecendo um caminho fácil e intuitivo de acesso a dados [BILLE, 1997] fazendo uso também da arquitetura cliente/servidor.

A *Web* mostra-se como a fonte de comunicação ideal para o compartilhamento de sofisticados controles e diversos dados, vindos de experimentos que fazem uso da instrumentação virtual, exatamente por oferecer uma interface homem-máquina intuitiva, de fácil manuseio. Proporciona a criação de ambientes virtuais que podem "imitar" aspectos do mundo real [ALLEN, 1998] possibilitando aos estudantes e pesquisadores apreciar a conseqüência de seus comandos, observando o estado inicial e final do aparato experimental.

Estão sendo desenvolvidos em diversas partes do mundo, ambientes interativos para monitoração de ensaios experimentais de forma remota, que não pretendem ocupar espaço apenas na área acadêmica, mas estender sua utilização para outras áreas que também necessitem de uma maior integração e troca de informações [BORGES, 1999].

Nos Laboratórios Virtuais pesquisados, existe a união da instrumentação virtual com as novas tecnologias de *hardware* e *software* para controle de equipamentos à

distância. Os ensaios experimentais são desenvolvidos baseando-se na instrumentação virtual para que o controle de instrumentos possa ser realizado através de um computador conectado diretamente aos instrumentos reais.

Um laboratório de medidas que pode ser conduzido via Internet, deve satisfazer dois requerimentos principais:

- O primeiro referente à utilização de sistemas de instrumentação de fácil uso e que sejam executados de forma eficiente para compartilhamento de equipamentos complexos e de alto custo. Neste caso o equipamento existe fisicamente e é compartilhado via rede de computadores.
- O segundo diz respeito a tornar a instrumentação disponível para acesso via computador, onde o instrumento é implementado e existe virtualmente, dispensando a presença do equipamento real, físico, funcionando próximo ao computador [BERTOCCO, 1998]. Por exemplo, um osciloscópio implementado no próprio computador funcionando como um osciloscópio físico.

2.5.1 Aplicação dos Laboratórios Virtuais: Área Médica

Na área médica o controle e o acompanhamento de processos de forma remota têm se destacado como, por exemplo, o sistema de monitoramento de sinais vitais onde o médico pode acompanhar a evolução do estado clínico de pacientes internados em uma UTI (Unidade de Terapia Intensiva) especial. Caso ocorra alguma alteração inesperada, o médico é avisado por e-mail ou outro meio de comunicação como Pager [REGGIANI, 2000]. Isso proporciona um menor desgaste do médico, que pode minimizar o tempo de deslocamento para visitar cada paciente, centralizando o controle, e possibilitando uma maior organização relacionada a cada paciente.

O acesso universal e a infraestrutura de rede, que pode facilitar o compartilhamento seguro de informações sobre pacientes e dados clínicos, tornam a Internet o meio ideal para a implementação de aplicativos visando a monitoração de pacientes geograficamente distantes do médico ou do ambulatório médico. Os sistemas

de tele-medicina baseados na *Web* [MAGRABI, 1999] incluem, por exemplo, serviços de monitoração do sistema cardíaco em tempo real. A importância deste tipo de monitoração é fundamental para a recuperação do paciente, uma vez que este pode estar longe do hospital, recuperando-se próximo aos familiares.

Para desenvolver o sistema voltado para área médica, referente à monitoração cardíaca via Internet, foram utilizadas tecnologias como *ActiveX* e **CGI** para implementação do aplicativos cliente. No lado servidor foi utilizado **CGI,** acrônimo de *Common Gateway Interface*, que define um caminho padrão por onde programas externos podem estabelecer comunicação com servidor *Web*. No cliente existem controles embutidos em página *Web* pelos quais o usuário/cliente pode ter acesso aos dados clínicos do paciente utilizando um *browser Web*. Este sistema possui restrição com relação ao *browser* e à plataforma operacional, uma vez que todo sistema é baseado nos *software*s da Microsoft (Windows 95, Windows NT, Internet Explorer, *Visual Basic*) para desenvolver os controles *ActiveX*.

2.5.2 Aplicação dos Laboratórios Virtuais: Educação a Distância

O desenvolvimento e uso de laboratórios virtuais onde estudantes e pesquisadores controlam instrumentos, muitas vezes específicos, de forma remota, onde existe uma metodologia de coleta e análise de dados, mostra-se como uma das recentes e promissoras atividades que une diferentes tecnologias para a elaboração de sistemas sofisticados e seguros.

Com as ferramentas (*software* e *hardware*) disponíveis no mercado, um novo modelo de ensino tem se destacado. Este modelo envolve o uso de redes de computadores, ensaios experimentais monitorados via computador, tornando as atividades experimentais disponíveis aos estudantes localizados a distância dos laboratórios convencionais.

O conceito de ensino a distância não é novo e tem sido amplamente explorado em diversas áreas devido aos avanços da tecnologia e novas oportunidades que surgem com a com o advento da Internet. Além do meio de comunicação promissor para todas as áreas, os avanços tecnológicos da instrumentação iniciam uma evolução dos velhos conceitos de ensino principalmente onde a teoria e a prática ficavam distantes [HOON, 1998]. O ensino tradicional envolve duas etapas: conceitos e prática. Nas classes tradicionais os conceitos são passados em aulas expositivas e reforçados em seções de laboratório onde os alunos têm a oportunidade de testar os conceitos assimilados [HESSELINK, 1999], [HOON, 1998], [SALZMANN, 1999]. Uma das aplicações que tem obtido destaque no meio educacional é o laboratório virtual como ferramenta de apoio na tentativa de melhorar a qualidade de ensino, incentivar o uso de novas tecnologias [FERRERO, 1999], [PASSERINI, 2000], [BENETAZZO, 1999] e atender a um maior número de alunos em aulas práticas.

É exatamente esta a proposta da maioria dos laboratórios virtuais pesquisados, ou seja, oferecer algo a mais, algo complementar ao ensino para que os alunos sejam beneficiados e possam usufruir das novas tecnologias desde a formação educacional. Existe uma dificuldade particularmente no ensino da engenharia elétrica e eletrônica pelo tempo necessário para elaborar, montar e testar experimentos [ARPAIA, 2000], ou seja, os problemas concentram-se no ensino e na prática experimental. A solução amplamente divulgada trata da disponibilização dos instrumentos reais para acesso remoto, baseado na arquitetura cliente/servidor. Há um *software* que foi implementado para controlar o acesso aos instrumentos no microcomputador que é o servidor de informação para o aluno geograficamente distante do aparato experimental.

O *AIM-Lab* (*Automated Internet Measurement Laboratory*) [SHEN, 1999], laboratório fisicamente localizado na Universidade Norueguesa de Ciência e Tecnologia, oferece um módulo do curso de caracterização de dispositivos e semicondutores para alunos de graduação, pós-graduação e pesquisadores.

A disponibilidade das alternativas de implementação oferece escolhas entre simplicidade, independência de plataforma e eficiência do sistema [SHEN, 2000]. Sendo assim, o sistema elaborado permite o uso eficiente de equipamentos laboratoriais nos

cursos presenciais ou a distância, relacionados à caracterização de dispositivos semicondutores com acesso remoto via *Web* [FIELDLY, 2000]. O AIM-Lab foi elaborado como um sistema baseado na arquitetura cliente/servidor utilizando a linguagem Java para a implementação do cliente e Visual C++ para o servidor. A linguagem Java oferece *design* flexível, sendo uma linguagem de programação conveniente para redes de computadores que necessita de portabilidade para tornar o sistema executável. Para o servidor, a linguagem escolhida foi o Visual C++ pois os *drivers* dos instrumentos utilizam funções em linguagem C, tornando a troca de comandos entre servidor e instrumentos uma tarefa fácil devido à integração das duas linguagens.

Para a execução dos experimentos no *AIM-Lab* nenhum arquivo específico precisa ser baixado da Internet pelo usuário. Qualquer usuário, através de uma página *Web* acessa o programa cliente implementado em Java. Os comandos gerados de acordo com o conjunto de parâmetros especificados pelo usuário são enviados ao servidor através do *socket TCP/IP*, acrônimo de *Transmission Control Protocol/ Internet Protocol* – protocolos utilizados para comunicação via Internet - implementados nos dois *software*s, cliente e servidor, para que a comunicação via Internet seja estabelecida. O *socket*, neste caso, representa um pequeno programa que é usado para comunicação entre processos de maneira bidirecional em sistemas de computação em redes locais e globais. O canal de comunicação criado com um *socket* pode ser como uma linha telefônica onde a conversa é bidirecional, e quando um fala o outro escuta e vice e versa [LINK 10]. Para evitar atrasos na comunicação, o servidor estabelece uma fila de usuários onde cada solicitação de usuário representa para o servidor um serviço a realizar e o local da fila só é desocupado quando o serviço é realizado e os dados são enviados ao cliente solicitante.

Na universidade de *Stanford* [LINK 11] há um laboratório virtual [HESSELINK, 1999] onde os alunos podem manusear instrumentos que não estão limitados à engenharia, estendendo as aplicações aos cursos de biologia, medicina, química, entre outras áreas relatadas na literatura. O enfoque ao laboratório desenvolvido está relacionado à forma de apresentação para que possam utilizar uma ferramenta didática

nos cursos oferecidos pela universidade [WALLER, 2000]. São destacados aspectos como a interatividade, a flexibilidade, enfim, os critérios para a elaboração de *software* com qualidade reconhecida fazendo uso de linguagens de programação que facilite a interação usuário-máquina via rede de computadores.

Na universidade de *Dalhousie* [PATON, 1998] é possível utilizar equipamentos de óptica sem a necessidade da presença física no ambiente experimental. O *Virtual Laser Lab* [LINK 4] permite que experiências que oferecem alto risco ao ser humano, sejam testadas de forma segura. Tudo é controlado com o auxílio de um *software* de controle que é distribuído pelo *browser Web*, em rede. Neste laboratório apenas o *browser Netscape Navigator* pode exibir o controle inserido no laboratório virtual desenvolvido pois são utilizadas as linguagens Java e JavaScrip. O sistema utiliza *LabVIEW*® e possui controle de uma câmera de vídeo disponibilizando imagens do ambiente em tempo real.

Na universidade Nacional de Singapura [SHIN, 2000], os alunos testam seus conhecimentos e a teoria assimilada com o auxílio de equipamentos e componentes podendo fixar os conhecimentos adquiridos durante as aulas teóricas [LINK 3]. Para o desenvolvimento foram utilizados como ferramentas de implementação do sistema as linguagens Java, *HTML* e o *software LabVIEW*® estabelecendo comunicação através da Internet via protocolo *TCP/IP* no esquema cliente/servidor já antes mencionado. Este laboratório possui também infra-estrutura para teleconferência disponibilizando imagens e som do ambiente de experimentação em tempo real [SALZMANN, 1998]. Esta alternativa é mais viável em ambientes onde há a utilização de meios de transmissão como fibra ótica, pois a transmissão de imagens em redes de grande extensão exige grande velocidade de transmissão proporcionada por meios fotônicos, o que encarece a implementação do sistema.

Com o aumento na largura de banda e uma maior difusão da Internet, não apenas a torna viável como sistema de educação a distância, como também deverá ser responsável pelo desenvolvimento desta “nova” metodologia de ensino nos próximos anos. Porém uma das deficiências existentes para cursos de educação a distância é a falta de aulas práticas em laboratório [KO, 2000]. O problema de implementação de um

Laboratório no sentido real da palavra, ou seja, um local onde se realiza labor (trabalho), torna-se relevante [FERNANDEZ, 2000].

Dentre as vantagens identificadas na prática do ensino a distância destacam-se: a possibilidade de acesso aos experimentos a qualquer hora do dia ou da noite, como forma de estimulação para que os alunos se interessem mais pelo tema da aula; e principalmente por seu uma forma interativa de aprendizado. As vantagens identificadas no ensino a distância compensam a ausência do professor, desde que ele esteja acompanhando os acontecimentos e caminhos que o novo curso está seguindo. Esta ausência pode ser suprida com o uso de salas de bate papo, *chats*, onde alunos e professores se "encontram" virtualmente para esclarecer dúvidas e discutir sobre assuntos pertinentes ao experimento realizado.

2.5.3 Aplicação dos Laboratórios Virtuais: Controle de Processos Industriais

O controle de processos industriais também pode ser feitos de forma remota utilizando basicamente a mesma tecnologia aplicada no desenvolvimento de um laboratório virtual voltado ao ensino a distância, ou seja, instrumentação, arquitetura cliente/servidor sendo o servidor responsável pelo controle efetivo do aparato experimental e o cliente sendo responsável pelo monitoramento via Internet.

2.6 O Laboratório Virtual deste trabalho

Este tópico tem como objetivo expor a definição e os aspectos relevantes deste projeto.

2.6.1 Definição

O Laboratório Virtual neste projeto refere-se ao ambiente que permite o acompanhamento e o controle de ensaios experimentais, que fazem uso da instrumentação virtual, de forma remota através da Internet, e se mostra como uma possibilidade de ambientes interativos para acesso e controle na forma real e virtual onde existe a disponibilização de ferramentas interativas desenvolvidas objetivando a relação homem-máquina [BORGES, 2000 b].

Como destaque e enfoque deste trabalho temos a integração da instrumentação virtual, com as novas tecnologias de *hardware* e *software* para a disponibilização de controles que podem ser acessados a distância, sem a necessidade de aplicativos específicos.

2.6.2 Objetivo

Este trabalho tem como proposta identificar uma forma de comunicação entre componentes de controle, instrumentação virtual e rede de computadores para que seja possível controlar e monitorar aparatos experimentais através de Intranet, em rede local, ou Internet, em rede remota.

Este ambiente que inclui ensaios experimentais, equipamentos de medida e sistema de controle remoto é apresentado como resultado deste trabalho. Com a evolução da tecnologia de *software*, surge a possibilidade de exportar o ambiente experimental de controle de instrumentos usando a Internet para que diferentes usuários possam compartilhar dados.

2.6.3 O sistema

O sistema para acessar aparato experimental via rede de computadores fazendo uso da instrumentação virtual com comunicação cliente/servidor possui como objetivos a conectividade, a interatividade, a transmissão de dados em rede, pelo menos um aparato experimental possibilitando a troca de dados, como descrito abaixo.

Conectividade – A implementação de um sistema de controle que permita o acesso remoto aos experimentos localizados em um laboratório físico (LME – Laboratório de Microeletrônica) instalado na Escola Politécnica da Universidade de São Paulo, via protocolo *TCP/IP* através da Internet.

Interatividade - O sistema deve permitir que o usuário interaja com software gerenciador da instrumentação virtual controlando os parâmetros necessários para realizar a experiência. O desenvolvimento de um sistema interativo representando uma interface entre usuários e aparato experimental independente da localização de ambos, usando como veículo de comunicação a Internet, se mostra como fator motivante para realização deste trabalho.

Transmissão de dados em Rede - Um protótipo de sistema baseado em um ambiente experimental envolvendo a transmissão de dados específicos relacionados ao experimento em questão através de uma rede de computadores. Estes dados incluem sinais e medidas de aquisição de dados ou alguma outra informação que dependerá exclusivamente da finalidade do experimento que esteja sendo executado. A intenção é desenvolver um sistema bastante interativo, no qual o usuário possa ter a opção real de controlar os parâmetros de um experimento físico também real. Utilizando uma arquitetura cliente-servidor, oferecida pelo *LabVIEW*®, é possível tornar realmente a experiência virtual. Também com o auxílio dos protocolos *TCP/IP* – para transferir os dados amostrados de ponta a ponta na rede; e o P1451 – protocolo padronizado para comunicação entre equipamentos de instrumentação virtual (sensores, atuadores, controladores).

Experimento – O objetivo éi elaborar um aparato experimental simples onde um cliente/usuário teria o controle de alguns parâmetros do ensaio, podendo fazer a leitura e análise de dados obtidos através de tabelas e gráficos que devem ser montados no decorrer da experiência. Serão desenvolvidos experimentos simples visando testar as técnicas para controle e compartilhamento de dados via rede de computadores para acesso através da Internet e *Intranet* com o auxílio de um *browser Web* capaz de exibir conteúdo dinâmico.

Dados – Os dados serão adquiridos por placas de aquisição da *National Instruments* utilizando o protocolo P1451 (protocolo entre equipamentos e microcomputador IEEE-P1451) e transferidos ao usuário via rede em um esquema cliente-servidor através do protocolo *TCP/IP*.

2.7 Considerações finais

Neste capítulo foram apresentados conceitos sobre sistemas que permitem o controle de instrumentação via Internet bem como uma descrição de alguns laboratórios virtuais pesquisados e as tecnologias que tornam possível a implementação de ambientes para o controle e monitoração de experimentos de forma remota via rede de computadores.

Como conclusão temos a descrição de um sistema ideal para um laboratório virtual envolvendo sistema distribuído de medição, e a infra-estrutura real, disponível na atualidade, diferente da ideal, apresentando limitações relacionadas ao número de centrais de processamento distribuídas na rede, envolvendo desempenho na transferência de informações devido ao intenso tráfego de dados em determinados horários.

O grande avanço tecnológico da Internet permite conectar pessoas distribuídas por todo o mundo, apresentando-se como motivação para o desenvolvimento de aplicações para popularizar o acesso a este meio de comunicação em diferentes áreas do conhecimento. Destaca-se o modelo de comunicação cliente/servidor utilizado pela maioria dos Laboratórios Virtuais pesquisados.

3. MATERIAIS E MÉTODOS

Na área de computação temos disponíveis a cada dia tecnologias mais sofisticadas que possibilitam implementações inovadoras de aplicativos cuja função é estabelecer comunicação em rede. Neste capítulo serão apresentados os materiais utilizados neste trabalho bem como a descrição da metodologia que tornou possível a implementação do Laboratório Virtual desenvolvido e que será apresentado no próximo capítulo.

3.1 Os materiais

Os materiais utilizados neste trabalho relacionam-se ao *hardware*, ao *software* e aos mecanismos para tornar possível a comunicação entre *hardware* e *software* em rede de computadores com acesso à Internet.

3.1.1 Os equipamentos - *Hardware*

Neste trabalho utiliza-se um microcomputador modelo PC (*personal computer*) com processador K6-II 500 MHz, 64 MB de memória RAM, disco rígido de 1GB, monitor, teclado e mouse. Desta configuração destaca-se a necessidade de grande quantidade de memória RAM e processador veloz (500MHz ou superior), pois todo processamento de informação é realizado neste computador.

Foi utilizada uma placa de aquisição de dados *DAQ* (*Data Acquisition*) modelo *AT MIO 16 X*® cuja função é obter os sinais gerados pelo circuito e estabelecer o interfaceamento entre os equipamentos de medida e o microcomputador. É necessário que haja compatibilidade entre a placa (*DAQ*), o *software* de gerenciamento da instrumentação virtual e o sistema operacional para garantir o funcionamento de todo o sistema elaborado [JOHNSON, 1994].

Um módulo *SCXI (signal conditioning extensions for instrumentation* – um condicionador de sinais) e arquitetura de instrumentação para automação e medidas. As vantagens na utilização do SCXI concentram-se na diminuição de ruídos de sinal e maior número de canais disponível devido a sua capacidade de multiplexação [NATIONAL INSTRUMENTS – 1998].

O circuito utilizado como material para este trabalho é ilustrado pela **Figura 7**. É possível identificar os canais de entrada além dos componentes envolvidos no circuito.

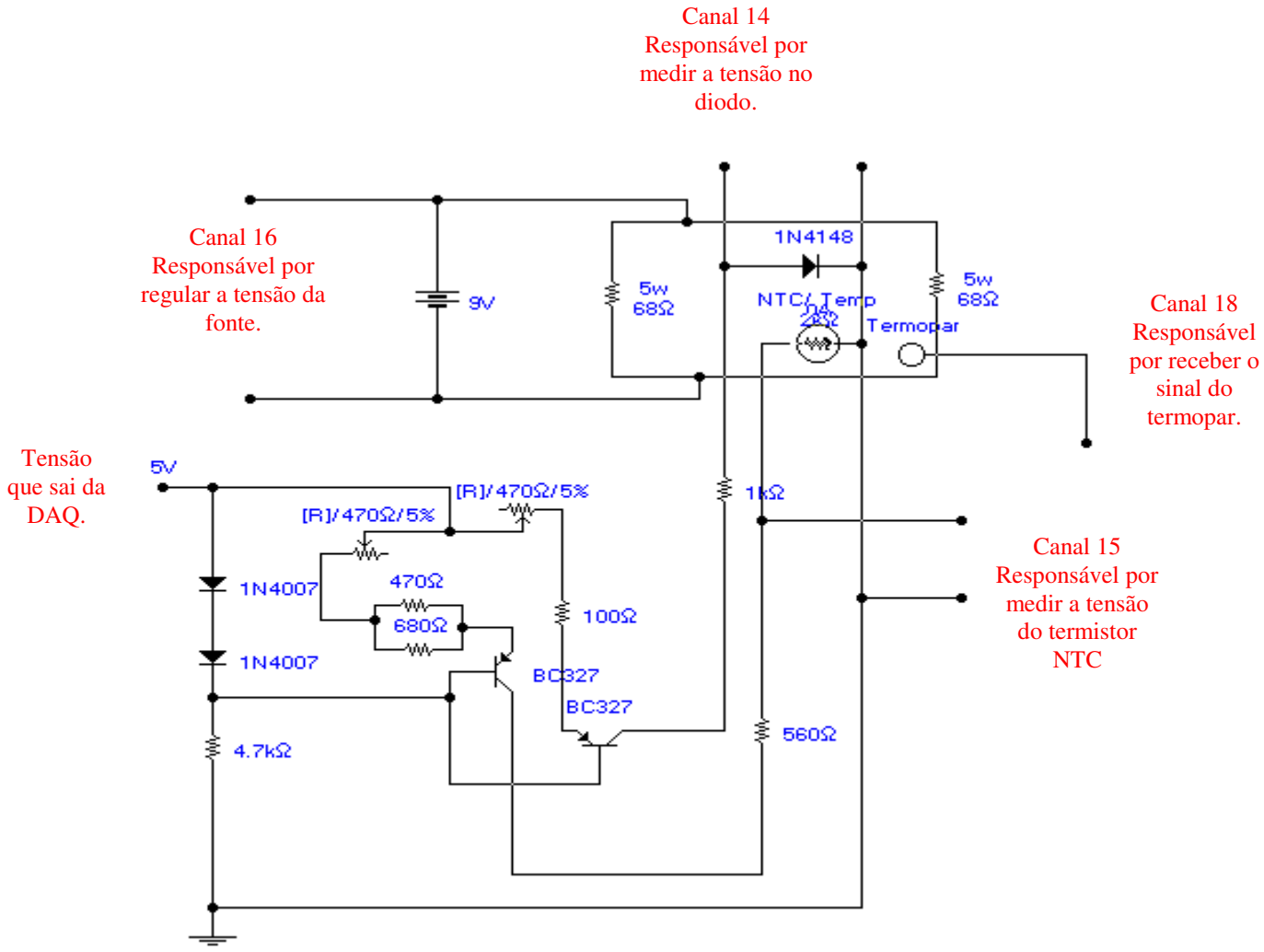

Figura 7 - Ilustração do circuito utilizado como material na parte experimental.

Alguns componentes são listados abaixo:

- Uma fonte de alimentação de 5 Volts;
- Um diodo 1N4003;
- Dois diodos 1N4148;
- Dois transistores BC 557 A;
- Dois potêciometros de 470 Ohms;
- Um resistor de 100 Ohms;
- Um resistor de 470 Ohms;
- Um resistor de 560 Ohms;
- Um resistor de 1 K Ohms;
- Um resistor de 3.3k Ohms.
- Um termistor.
- Um termopar tipo J.

3.1.2 Os programas utilizados - *Software*

Relacionado à área de instrumentação virtual encontramos tecnologias de *software* que abrangem desde a aquisição até o compartilhamento de informações obtidas de ensaios experimentais. Para a implementação destes ensaios é necessário elaborar uma experiência que possa ser controlada através da instrumentação virtual [CHEN, 1999 b]. O *software* que tem obtido destaque na execução de tarefas como controle de aparatos experimentais é o *LabVIEW®* [LINK 7].

O *LabVIEW®* é a ferramenta que mais satisfaz os requisitos para o desenvolvimento de controle local de experimentos oferecendo ambiente de programação e análise sofisticado (descrito no capítulo anterior) para sistema operacional *Windows* [CHEN, 1999 a], [STEGAWSKI, 1998]. O *LabVIEW®* proporciona a elaboração de sistemas para o compartilhamento de instrumentos de alto custo com outras universidades e centros de pesquisa através da *Web* [PALOP, 2000].

O *DataSocket®* é uma tecnologia e uma coleção de ferramentas desenvolvidas pela *National Instruments* para facilitar a troca de dados e informações entre uma

aplicação e um número diferente de fonte de dados e destinatários em rede. Com o *DataSocket®* é possível especificar as fontes de dados e destinatários para conexões usando *URL's* (*uniform resources locators*) que prendem-se ao modelo URL familiar (por exemplo, http://...). Esta ferramenta pode ser utilizada no desenvolvimento de VI's (*virtual instruments*) em *LabVIEW®*, bem como no desenvolvimento de programas em ambientes de programação como *Delphi*, *Visual Basic*, *Visual C++* e oferece um servidor *DataSocket Server®* que permite a troca de informações entre aplicativos desenvolvidos em linguagens de programação distintas, como *LAbVIEW®* e *Visual Basic*, por exemplo.

O *DataSocket Server®* é o programa que recebe e envia os dados do experimento para o cliente e/ou vice-versa. Ao implementar o programa controlador do experimento utilizando *LabVIEW®*, devem ser usadas as sub-rotinas que permitem o envio/recebimento de dados para um determinado computador cujo endereço deve seguir o padrão *URL*. Este endereço deve ser o *IP* do computador onde o *DataSocket Server®* está instalado. Ao receber os dados do experimento, ou seja, do programa que se comunica e controla a experiência através da instrumentação virtual, o *DataSocket®* envia para os programas clientes que estão ligados em rede, ou seja, os clientes não recebem os dados diretamente do programa que controla a experiência. A **Figura 8** ilustra o esquema de comunicação utilizando o *DataSocket®*.

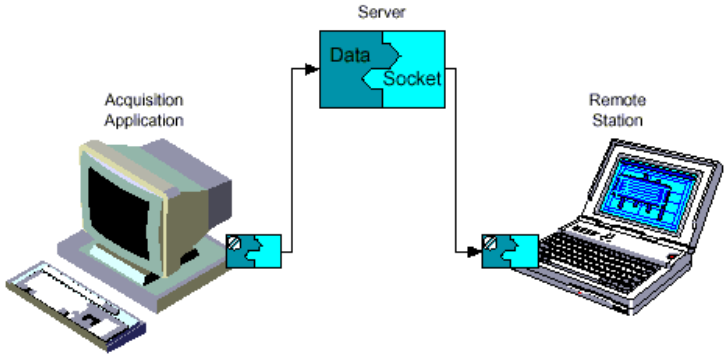

Figura 8 - ILUSTRAÇÃO DO ESQUEMA DO FUNCIONAMENTO DO *DATASOCKET SERVER®*. SOBRE A ARQUITETURA cliente/servidor.[3]

[3] Figura retirada do Application Note nº. 127 - National Instruments.- Junho 1999.

O *ComponentWorks®*, conjunto de ferramentas para ambiente de programação como *Visual Basic*, *Visual C++* e *Delphi*, oferece os componentes necessários para o desenvolvimento de controles *ActiveX*. Em um primeiro estágio estes componentes são criados em ambiente de programação como *Visual Basic*. Em um segundo estágio o controle *ActiveX* criado deve ser inserido em uma página *HTML* para que possa ser visualizado em um *browser Web* disponibilizando, desta forma, os controles de equipamentos para que qualquer cliente conectado à Internet possa visualizar o painel de controle e ter acesso aos equipamentos de forma virtual.

O *ComponentWorks®* é uma coleção de controles *ActiveX* para aquisição, análise e apresentação de dados sem a necessidade do aplicativo container (local onde o controle *ActiveX* é desenvolvido e implementado) do *ActiveX*.

Os controles *ActiveX* [NATIONAL INSTRUMENTS, 1999] do *ComponentWorks®* utilizados neste trabalho foram implementados em *Visual Basic*. Os componentes do pacote *ComponentWorks®* usados para este trabalho são os controles *DataSocket®* , que consistem em ferramentas e controles *ActiveX* para compartilhar dados entre aplicações de um número diferentes de fonte de dados ou destinos, incluindo outras aplicações, arquivos, e servidores *Web* e *FTP* (*file transfer protocol* – protocolo de transferência de arquivos).

É necessário implementar páginas *Web* para divulgação de informações sobre o Laboratório Virtual disponibilização dos experimentos para que se possa exibir dados em qualquer *browser* (padrão *Windows* como *Internet Explorer* ou *Netscape Communicator*), utilizando a linguagem *HTML* (*HyperText Markup Language*). O usuário poderá ter acesso aos dados (controles de equipamento) utilizando seu *browser* de navegação na Internet.

O sistema desenvolvido neste trabalho funciona no sistema operacional Windows 95, padrão compatível com o gerenciamento e controle implementado.

3.2 Etapas pré-estabelecidas para a realização do trabalho

A primeira etapa deste trabalho consiste na escolha de uma experiência que possa ser utilizada didaticamente envolvendo comunicação entre dispositivos de medida, microcomputador e rede de computadores (local e remota). A experiência deste trabalho foi escolhida por oferecer simplicidade e aplicabilidade em aulas didáticas de eletrônica está relacionada à caracterização de um diodo e um termistor submetidos a diferentes temperaturas.

Definida a experiência é necessário elaborar o controle via computador. Esta etapa pode ser elaborada com o auxílio do *software LabVIEW®* mencionado anteriormente. A próxima etapa consiste na elaboração dos controles para funcionamento em rede local de computadores também com o auxílio do *software LabVIEW®*.

A quarta etapa diz respeito a utilização de controles *ActiveX* para que o painel de controle já elaborado em *LabVIEW®* possa ser visualizado através de um *Web browser* comercial.

Estando todo o sistema funcionando, ou seja, estando todos os programas implementados trocando dados via rede local e rede remota surge a necessidade de disponibilizar os controles via página *HTML*. Nesta fase deve ser construído um sítio na Internet para testar os controles implementados.

Após os últimos ajustes do sistema o mesmo deve ser disponibilizado para toda comunidade Internet, para que qualquer pessoa, de qualquer lugar do mundo, possa ter acesso ao sistema implementado. O diagrama de blocos ilustrado na **Figura 9** apresenta as etapas a serem cumpridas desde a idealização até a conclusão deste trabalho.

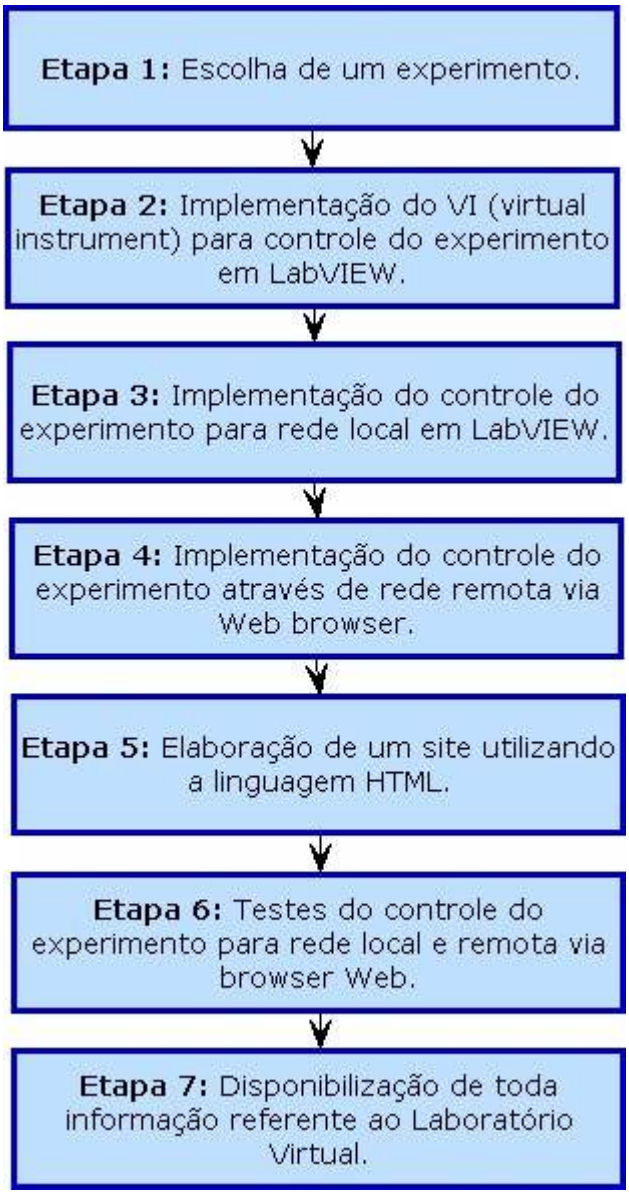

Figura 9 - Etapas a serem cumpridas para a realização deste trabalho.

3.3 A metodologia adotada

A implementação de sistemas de controle para Internet faz uso da arquitetura cliente/servidor. O servidor, situado próximo ao experimento, comunica-se com este transferindo e recebendo os comandos vindos do cliente para o experimento através da

interface de comunicação (porta paralela, porta serial, GPIB). O *VI* desenvolvido no *LabVIEW®* consegue transferir e receber sinais elétricos dos equipamentos de medida.

Para que o cliente e o servidor se comuniquem é necessário haver um protocolo de comunicação para rede. Existem vários protocolos, sendo os mais utilizados para transporte de dados em rede o *UDP* (*User Datagram Protocol*), *TCP* (*Transmission Control Protocol*) e *IP* (*Internet Protocol*).

Estes últimos, TCP e IP, serão utilizados neste projeto devido a segurança que oferecem ao transferir os dados na rede de comunicação. TCP, embora mais lento que UDP, oferece garantia de que os dados transferidos no processo servidor/cliente chegaram ao destino de forma correta, pois executa um teste, o que não ocorre no transporte através do UDP.

Para o controle do experimento (envolvendo sensores), instrumentos virtuais (*VI – virtual instrument*) serão implementados em *LabVIEW®*. Baseados na arquitetura cliente/servidor o *VI* de controle denominado servidor do experimento será o responsável pela comunicação com os equipamentos envolvidos, enviando e recebendo dados para testes. Também será responsável por controlar o acesso de usuários delimitando um tempo pré-determinado para a execução do experimento.

A escolha da tecnologia *ActiveX* e da linguagem de programação *Visual Basic* se deve ao fato de ambas serem inteiramente compatíveis com o *software* para instrumentação, *LabVIEW®*, além da facilidade de implementação atendendo ao requisito de menor tempo de desenvolvimento. Estas ferramentas estão totalmente disponíveis para o que se propõe desenvolver neste projeto. A tecnologia *ActiveX* [LINK 12] oferece velocidade de transferência, uma vez que compacta os dados antes do empacotamento para ser transmitido em rede, diferente da tecnologia Java. Além de ser fácil de usar, oferece acessibilidade e interface amigável [LINK 13] e por isso será utilizada neste projeto.

Os controles *ActiveX* são componentes de *software* compilados na tecnologia COM. Essencialmente são programas modulares cujo *design* deve oferecer funcionalidade na execução dos aplicativos dentro de um *Web browser*, executar tarefas,

computar informações e permitir a comunicação via arquitetura cliente/servidor [MAGRABI, 1999].

Controles *ActiveX* são os objetos interativos em uma página da *Web* que fornece funções interativas controláveis pelo usuário. Documentos de *ActiveX* permitem que os usuários tenham acesso a documentos diferentes dos escritos em *HTML* e permitem a implementação de conteúdos dinâmicos para páginas [FARRAR, 1997], [LINK 14].

O *Visual Basic* é uma das linguagens possíveis para a utilização do *ComponentWorks®*, da *National Instruments*.

Um servidor *WWW* é fundamental, pois deixa disponível para todos os usuários da Internet o *software* aplicativo em questão através de páginas implementadas em *HTML* e a interface com o cliente que será desenvolvida em *ActiveX*. O protocolo de comunicação padrão da Internet é o *TCP/IP*, que será utilizado como protocolo de comunicação do sistema que a ser desenvolvido.

A estrutura do sistema é ilustrada na **Figura 10**, onde é possível identificar as partes envolvidas, ou seja, os instrumentos, o servidor e o cliente estabelecendo comunicação via Internet. A **Figura 11** representa a metodologia de desenvolvimento adotada para este trabalho, apresentando os caminhos possíveis.

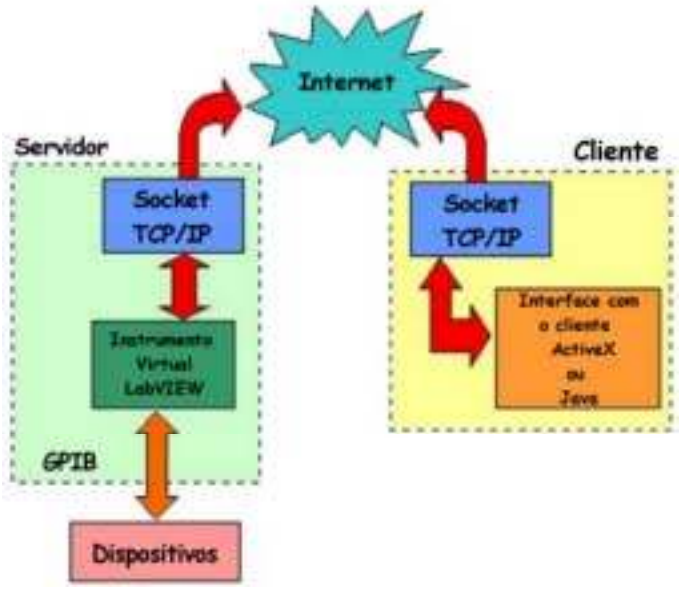

Figura 10 - Estrutura esquemática da comunicação do todo o sistema via arquitetura cliente/servidor.

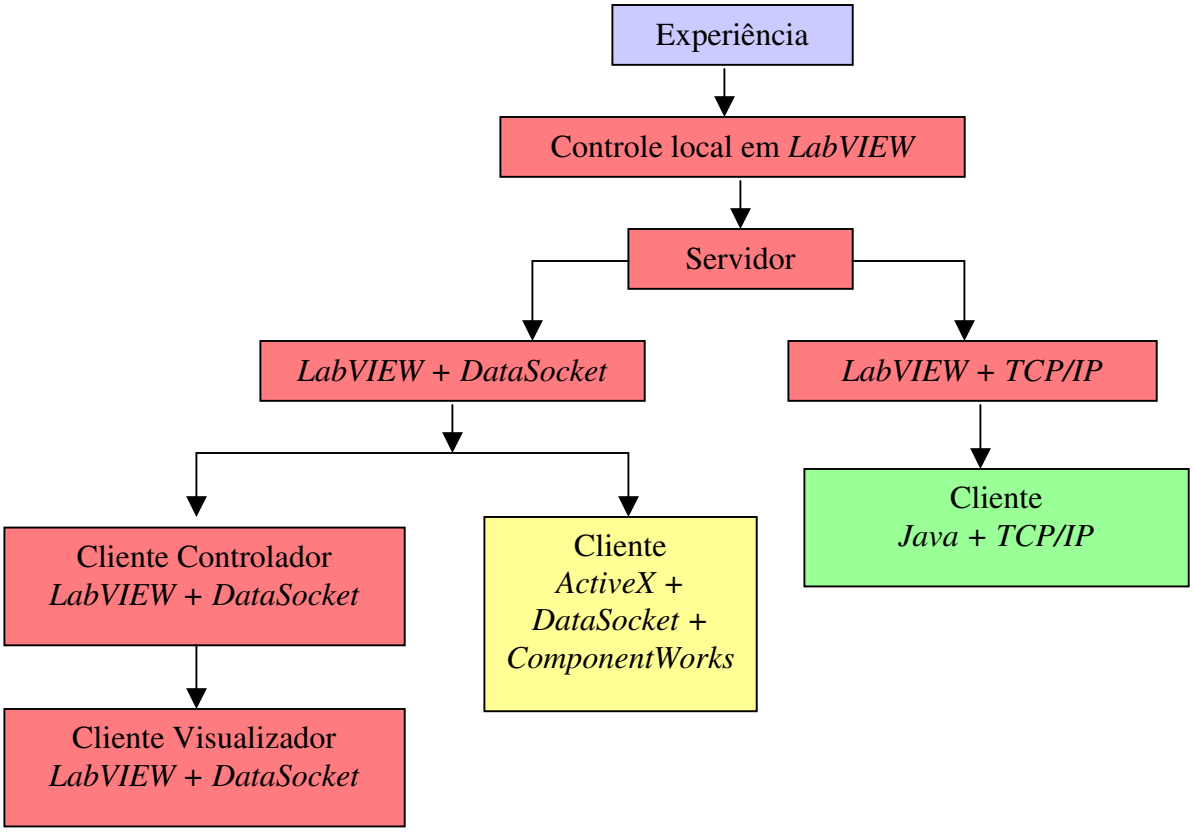

Figura 11 - Metodologia de desenvolvimento adotada.

3.4 Motivação para o uso do *ActiveX*

Em princípio, a tecnologia utilizada para implementar os controles da instrumentação virtual de forma a ser exibida embutida em um navegador *Web* é o *ActiveX*.

Para exportar o painel de controle dos instrumentos desenvolvidos, baseando-se na arquitetura cliente/servidor, é necessária a implementação de uma interface para o controle à distância com o auxílio da tecnologia *ActiveX* . A implementação do controle local foi elaborada com o auxílio do *LabVIEW*®, *software* que proporciona a

instrumentação virtual. A comunicação entre *ActiveX* e *LabVIEW®* via Internet se dá pelo uso do protocolo de comunicação *TCP/IP*.

A tecnologia *ActiveX* [FARRAR, 1997] define uma nova especificação permitindo que os controles sejam menores e mais eficientes, oferecendo benefícios como:

- Menor tempo de desenvolvimento, dispensando um ambiente de programação específico, ou seja, pode ser implementado em ambientes de programação com *Visual Basic*, *Delphi*, etc;
- É uma tecnologia completamente compatível com *LabVIEW®*;
- Pode ser utilizado em qualquer aplicativo *Microsoft*, como por exemplo no *Excel*.

A *Microsoft*, empresa que desenvolveu a tecnologia, disponibiliza gratuitamente um *software* para desenvolvimento de controles *ActiveX*.

Uma desvantagem na utilização dessa tecnologia reside no fato de ser compatível apenas com o *browser* desenvolvido pela *Microsoft*, o *Internet Explorer.* Isto não chega a ser um ponto preocupante uma vez este *browser* vem instalado juntamente com sistema operacional *Windows* não dificultando o acesso dos usuários ao Laboratório Virtual.

3.5 Considerações Finais

Este capítulo descreveu os materiais e a metodologia utilizada para o desenvolvimento deste trabalho destacando a importância e justificando as escolhas efetuadas. Ao final encontra-se o diagrama de blocos que ilustra as etapas do projeto com uma breve descrição do que foi realizado em cada uma delas.

O próximo capítulo apresentará os resultados obtidos bem como as conclusões sobre cada etapa do trabalho.

4. RESULTADOS EXPERIMENTAIS

Neste capítulo são descritos os resultados obtidos na implementação do sistema que permite o controle de instrumentação virtual via Internet bem como as fases de desenvolvimento do Laboratório Virtual, envolvendo a escolha da experiência e a implementação do software de controle remoto embutido em *browser Web*.

4.1 Ensaio experimental

Neste trabalho foi utilizado um circuito elétrico elaborado para permitir a caracterização elétrica de um diodo e um termistor em função da temperatura. Foi montado um bloco composto por duas resistências com capacidade de dissipação de potência de 5 W cada uma para gerar calor e conduzir ao diodo e ao termistor através de diferentes regimes de temperatura de operação. A situação térmica deste bloco é registrada por meio de um termopar tipo J composto por Ferro – Constantan. A condição operacional do Diodo e do Termistor é ajustada por uma fonte de tensão controlada e um circuito transistorizado que permite regular a corrente de polarização em cada componente. O ajuste fino do valor dessa corrente de polarização é obtido por meio dos potenciômetros.

O objetivo desta experiência é oferecer ao usuário um ambiente instrumental operacional para efetuar o estudo do efeito da temperatura sobre o comportamento de um diodo e um termistor. O ensaio permite conduzir ao diodo e termistor a diferentes condições de trabalho de modo a obter, nas curvas características de funcionamento, o efeito provocado pela temperatura ambiente.

Um dos métodos principais para medida elétrica de temperatura explora a mudança da resistência elétrica de certos tipos de materiais metálicos. Neste caso, o princípio da técnica de medida consiste em colocar o dispositivo sensível a temperatura

em contato com o ambiente no qual se deseja medir a temperatura. Assim, a medida de sua resistência indica a temperatura do dispositivo e conseqüentemente do ambiente.

Para fins experimentais é conveniente utilizar a resistência R, que depende do elemento resistivo em questão e se relaciona com resistividade ρ através das grandezas geométricas do dispositivo tais como o comprimento ***l*** e a secção transversal ***A*** do elemento [MEYIR, 1994].

$$R = \rho \frac{d}{A} \tag{01}$$

Portanto, para um dado elemento resistivo metálico, tem-se que a resistência R aumenta com a temperatura T, em conseqüência da variação que sofre a resistividade ρ. Logo, feito um ajuste de escala, a curva do gráfico de R por T é essencialmente a mesma que a da **Figura 12**.

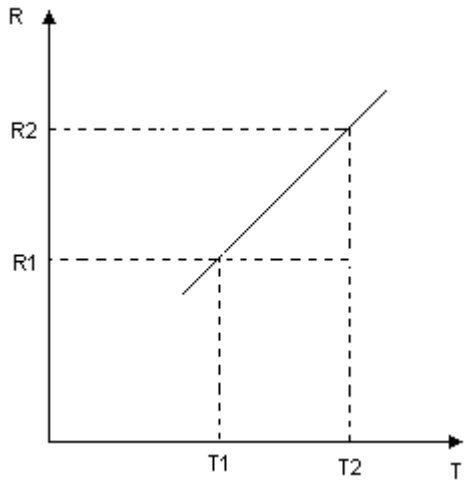

Figura 12 - Variação da resistência em função da temperatura no trecho linear.

O *coeficiente de temperatura* α_{T1} é definido como:

$$\alpha_{T1} \equiv \frac{\tan\alpha}{R_1} = \frac{R_2 - R_1}{R_1}\frac{1}{T_2 - T_1} \tag{02}$$

Usualmente, toma-se como referência a temperatura T_1=20 ºC, de modo que $\alpha_{T1}=\alpha_{20}$. Assim, tem-se uma expressão que permite calcular a resistência de um

elemento a uma certa temperatura T_2 (em ºC), desde que sejam conhecidos seu coeficiente de temperatura e a resistência na temperatura de referência, R_{20}:

$$R_{T2} = R_{20}\left[1 + \alpha_{20}\left(T_2 - 20\right)\right] \tag{03}$$

Em termos práticos, quanto maior o coeficiente de temperatura, maior é a sensibilidade do dispositivo com as mudanças de temperatura.

De modo geral, espera-se que as curvas características dos elementos estudados sofram alterações conforme a modificação da temperatura. A curva característica de um elemento resistivo pode ser representada pela variação de tensão em função da intensidade de corrente.

4.1.1 Resistor

Da relação linear entre a resistividade e a resistência, demonstra-se que a resistência depende da temperatura, de um modo também linear.

O comportamento da *curva característica* do resistor fica descrito pela *Lei de Ohm*, que estabelece uma relação linear entre a diferença de potencial no elemento e a corrente, já que a resistência é um fator constante [MEYIR, 1994]. Sendo assim, quando existem variações na temperatura, verifica-se que a resistência aumenta linearmente, e visto que na curva característica a resistência é descrita pela inclinação da reta no gráfico I x V, concluí-se que o efeito produzido por tais variações nas curvas estudadas diz respeito ao coeficiente angular.

4.1.2 Diodo

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

$$I = I_0\left(e^{V/\eta V_T} - 1\right) \tag{04}$$

Um valor positivo de I significa que a corrente flui do lado p para o lado n. O diodo está polarizado diretamente se V é positivo, indicando que o lado p da junção é

positivo com relação ao lado n. O coeficiente η tem um valor próximo de 1 para valores de tensão compreendidas no intervalo de 500mV a 700mV .

O símbolo V_T= KT/q representa a tensão equivalente da temperatura (potencial termodinâmico) e é dado pela equação [MEYIR, 1994]:

$$V_T = \frac{T}{11600} \tag{05}$$

a temperatura ambiente (T = 300 K), V_T = 0,026 Volts ou V_T = 26 mV. A **Figura 13** mostra o gráfico das curvas características de um diodo submetido a diferentes temperaturas.

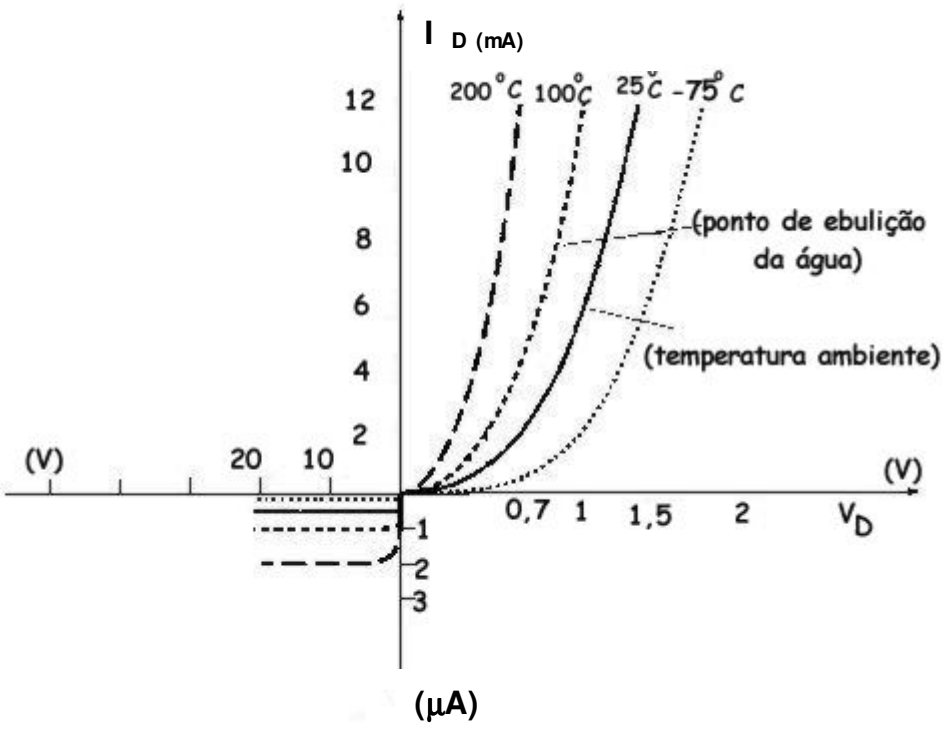

Figura 13 - Curvas características de um diodo a diferentes temperaturas.

4.1.3 Termistor

Os termistores são resistores sensíveis a temperatura fabricados de material semicondutor, tais como óxido de níquel, cobalto, ou magnésio e sulfeto de ferro, alumínio ou cobre. Óxidos semicondutores, diferente dos metais, pode exibe uma resistência que decresce com a temperatura, são os chamados NTC (do inglês, negative temperature coeficiente) [MEYIR, 1994]. A relação para um termistor deste disso pode ser expressa por

$$\ln (R/R_0) = \beta(1/T - 1/T_0) \qquad (06)$$

ou

$$R = R_0 \exp[\beta(1/T - 1/T_0)] \qquad (07)$$

onde

R é a resistência do termistor na temperatura T
R_0 é a resistência do termistor na temperatura T_0
β é a constante do material (3000 - 5000 K)

A sensibilidade S do termistor é obtida da equação (07) como

$$S = \Delta R/(R.\Delta T) = -\beta/T^2 \qquad (08)$$

A equação (07) indica que a resistência R de um termistor decresce exponencialmente com a temperatura. Uma curva de resposta típica de um termistor é mostrada na **Figura 14**.

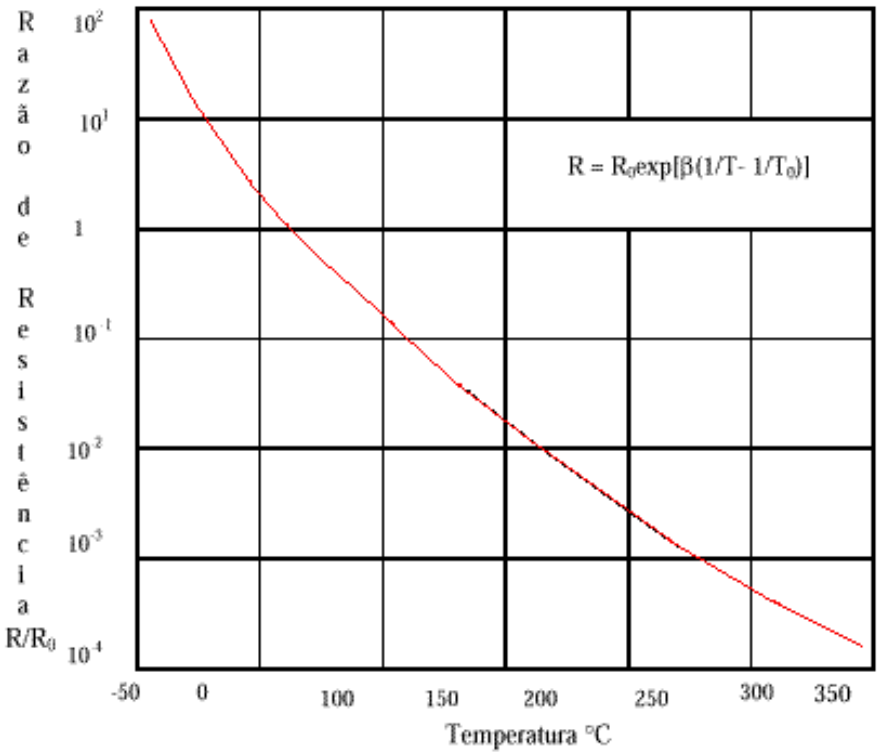

Figura 1.3) Resistência como função da temperatura para termistor tipo NTC

Figura 14 - Resistência como função da temperatura para termistores tipo NTC.

4.2 Primeira Etapa - Sistema de Instrumentação Virtual

Para dar início ao desenvolvimento do sistema de controle remoto, onde parâmetros informados pelo usuário estão envolvidos na caracterização de um diodo e um termistor submetidos a diferentes temperaturas, foi elaborado um VI (virtual instrument) com um único e exclusivo objetivo: obter os valores de tensão de cada componente e a temperatura de um termopar associado ao circuito elétrico.

O painel de controle mostrado na **Figura 15** ilustra a interface desenvolvida. Em cada campo é mostrado o valor de tensão lido bem como o valor da temperatura em Graus Celcius.

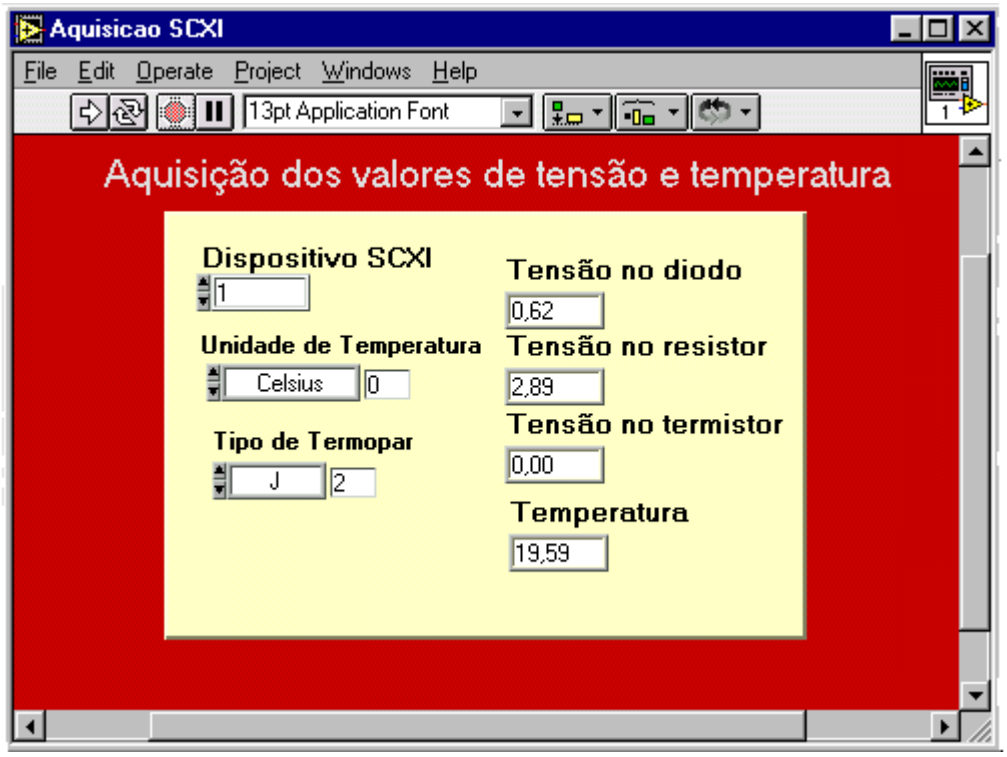

Figura 15 - Interface do VI-Aquisição SCXI.

No diagrama de blocos mostrado na **Figura 16** é possível identificar as funções e os canais utilizados para a leitura de cada valor. A aquisição de dados começa quando o usuário informa os parâmetros (tensão e tempo de amostragem), ou seja, este programa servirá como um SubVI onde a entrada será o valor de tensão inicial e as saídas serão os valores de tensão lidos a partir dos componentes.

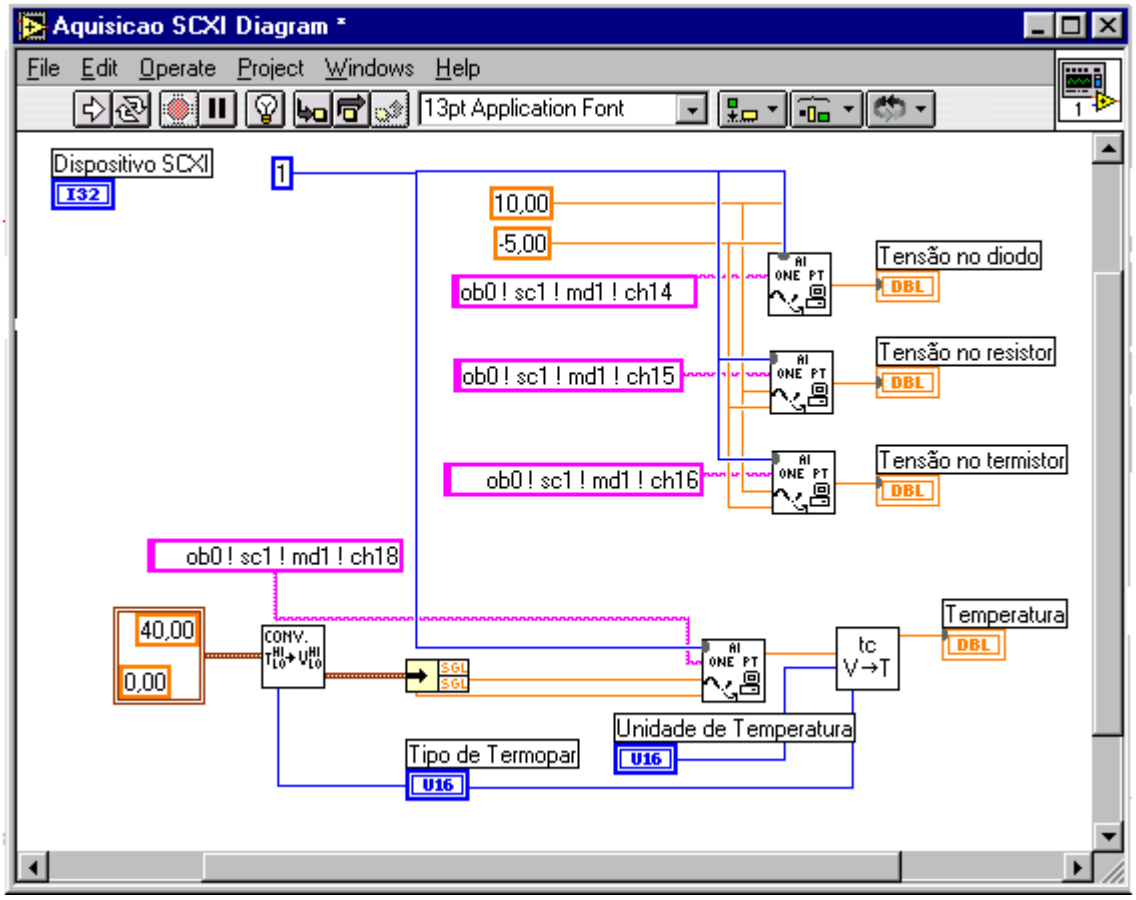

Figura 16 - Diagrama de blocos do VI-Aquisição SCXI.

Nesta etapa de implementação o sistema cliente/servidor elaborado é composto por três VIs distintos cada um com uma função específica denominados: VI-Servidor (que funciona em conjunto com o *DataSocket Server*® da *National Instrument*), o VI-Controlador e o VI-Visualizador. Cada programa desenvolvido será descrito nos tópicos a seguir. Nesta fase todos os aplicativos foram desenvolvidos em *LabVIEW*®.

4.2.1 O VI-Servidor

As principais funções do programa VI-Servidor são:

- Estabelecer a comunicação com o *DataSocket Server*®;

- Estabelecer a comunicação com os equipamentos de medida através de uma placa de aquisição de dados;
- Realizar a aquisição de dados de acordo com os parâmetros vindos do VI-Controlador repassados pelo *DataSocket Server®* bem como oferecer os parâmetros resultantes para que o *DataSocket Server®* envie os valores de tensão obtidos na aquisição de dados para o cliente solicitante, ou seja, para o VI-Controlador que solicitou a monitoração do aparato experimental.
- Gerenciar os acessos dos usuários liberando o controle apenas para um VI-Controlador por vez e enviar os dados resultantes para todos os VI-Visualizadores existentes. Sendo assim, todos os usuários que solicitam a visualização dos dados resultantes são atendidos e são liberados tão logo quanto termine a experiência.

A interface do VI-Servidor pode ser observada na **Figura 17**. É possível acompanhar apenas a comunicação cliente/servidor e, desta forma, identificar se a experiência está sendo realizada ou não. Como o intuito é controlar os equipamentos remotamente não foram disponibilizadas entradas de dados diretamente no VI-Servidor.

Figura 17 - Interface do VI-Servidor implementado em *LabVIEW®* responsável pela comunicação entre o aparato experimental e o *DataSocket Server®*.

É possível identificar na interface do **VI-Servidor** informações que indicam o que está acontecendo desde a recepção da solicitação de controle do experimento até a finalização quando são enviados ao cliente solicitante os dados resultantes da experiência. São informados também os estados de comunicação com o *DataSocket Server*® e os valores enviados pelo **VI-Controlador**.

O fluxograma que descreve as tarefas executadas pelo VI-Servidor é apresentado na **Figura 18**, onde é possível identificar cada passo que o programa servidor irá realizar.

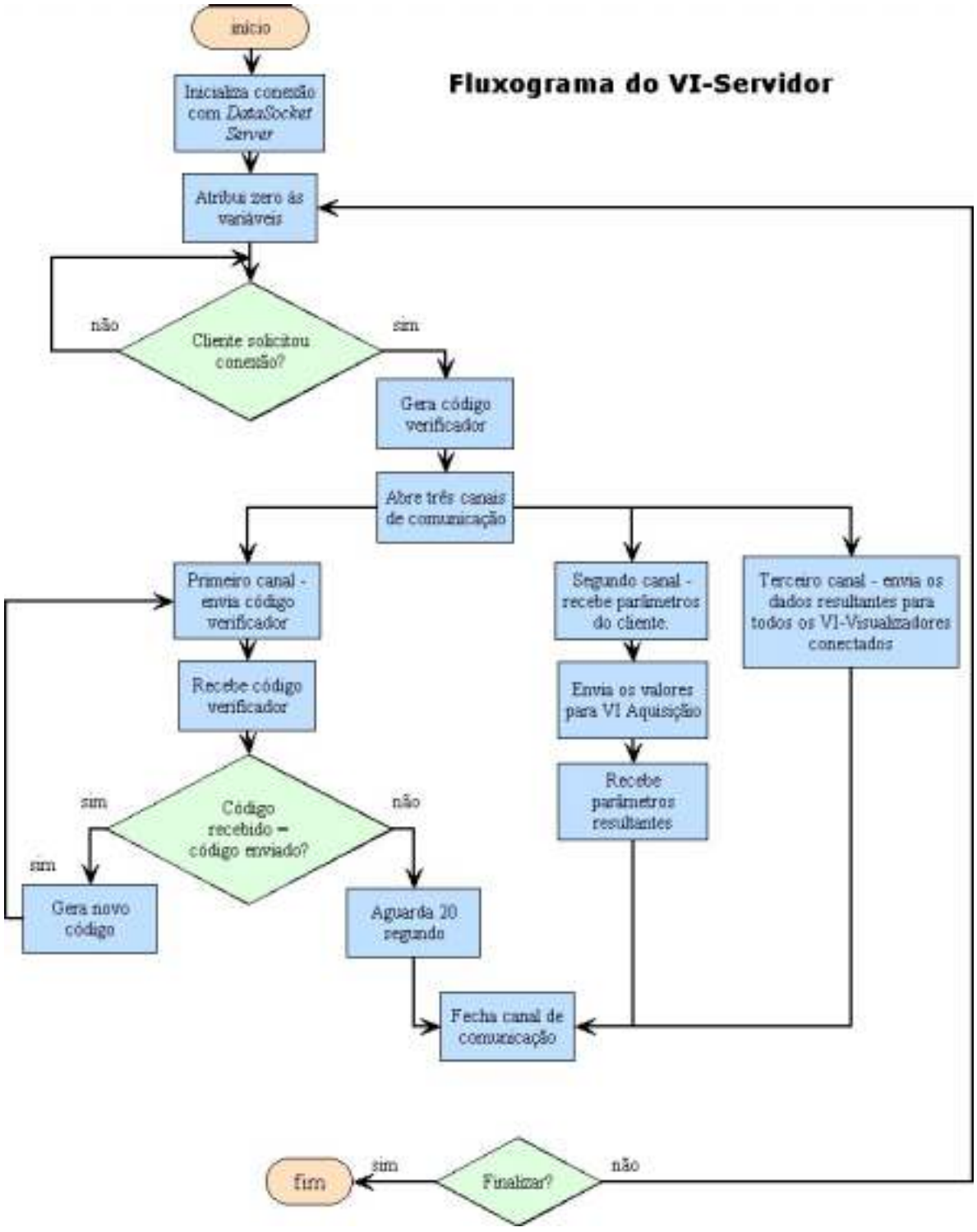

Figura 18 - Fluxograma que descreve o funcionamento do programa VI-Servidor **desenvolvido em *LabVIEW®*.**

4.2.2 O VI-Controlador

O VI-Controlador, cuja interface é apresentada na **Figura 19**, é o responsável por estabelecer a comunicação com o *DataSocket Server*® e enviar os parâmetros necessários para a realização da experiência, ou seja, tem como finalidade enviar e receber os parâmetros para que a experiência seja acionada e as medidas realizadas.

Figura 19 - Interface do VI-Controlador implementado em *LabVIEW*® responsável por controlar a experiência de forma remota.

Para que a experiência possa ser controlada o usuário deverá configurar os parâmetros da experiência nos seguintes campos: Intervalo de Tempo entre Amostras em segundos; Tensão Inicial em Volts; Tensão Final em Volts e Tensão de incremento em Volts. Após configurá-la basta iniciar o processo através do botão Enviar.

Ao iniciar o processo o **VI-Controlador,** o cliente, passará a enviar e receber uma série de códigos do servidor implementados para garantir que a conexão entre o VI-Servidor não foi perdida, ou seja, para testar se o canal de comunicação continua aberto. Os campos Nome do Servidor e Status Canal Requisição indicam o nome do Servidor onde está localizado o **VI-Servidor** e o estado de comunicação no início da conexão.

Foram inseridos na interface "*Leds"* (Esperando Resposta, Enviando Dados, Experiência em Execução e Cancelado) que indicam o que está ocorrendo na execução da troca de dados, ou seja, informa o estado (*status*) da comunicação. O primeiro "*Led"* indica que o **VI-Controlador** está enviando um pacote solicitando a execução da experiência. Este permanecerá ligado até que o programa receba um pacote de dados autorizando o controle. O segundo "*Led"* indica quando o **VI-Controlador** está enviando os dados e o terceiro "*Led"*, que é independente dos outros, é acionado caso o **VI-Servidor** já esteja recebendo informações de outro **VI-Controlador**, ou seja, indica se outro usuário está controlando a experiência no momento. O *led* Cancelado é acionado quando o cliente cancelar a experiência e assim permanece até que o **VI-Servidor** reconheça o comando de Cancelamento da experiência e libere a experiência para o próximo usuário que solicitar o controle.

O botão Cancelar é responsável pelo envio de solicitação de cancelamento da experiência, ou seja, caso o usuário pressione este botão, o programa **VI-Controlador** enviará um código para o servidor requisitando o cancelamento da experiência, e recebido esse código o servidor reenvia ao cliente um pacote confirmando o seu recebimento. Portanto ao apertar o botão Cancelar indica o inicio do processo para cancelamento da experiência. Tal rotina foi implementada para garantir que o cliente não se desconecte sem que o **VI-Servidor** tenha realmente cancelado a experiência.

O princípio de funcionamento do **VI-Controlador** leva em consideração que o **VI-Servidor** será responsável apenas por adquirir os dados da experiência, ou seja, será executada a aquisição de informações caso o **VI-Controlador** envie os parâmetros. Ao final do controle o cliente, que possuía a prioridade de controle do aparato experimental é desconectado liberando o acesso ao próximo **VI-Controlador** que solicite informações enviando os parâmetros.

O fluxograma que descreve as tarefas executadas pelo **VI-Controlador** é apresentado na **Figura 20**.

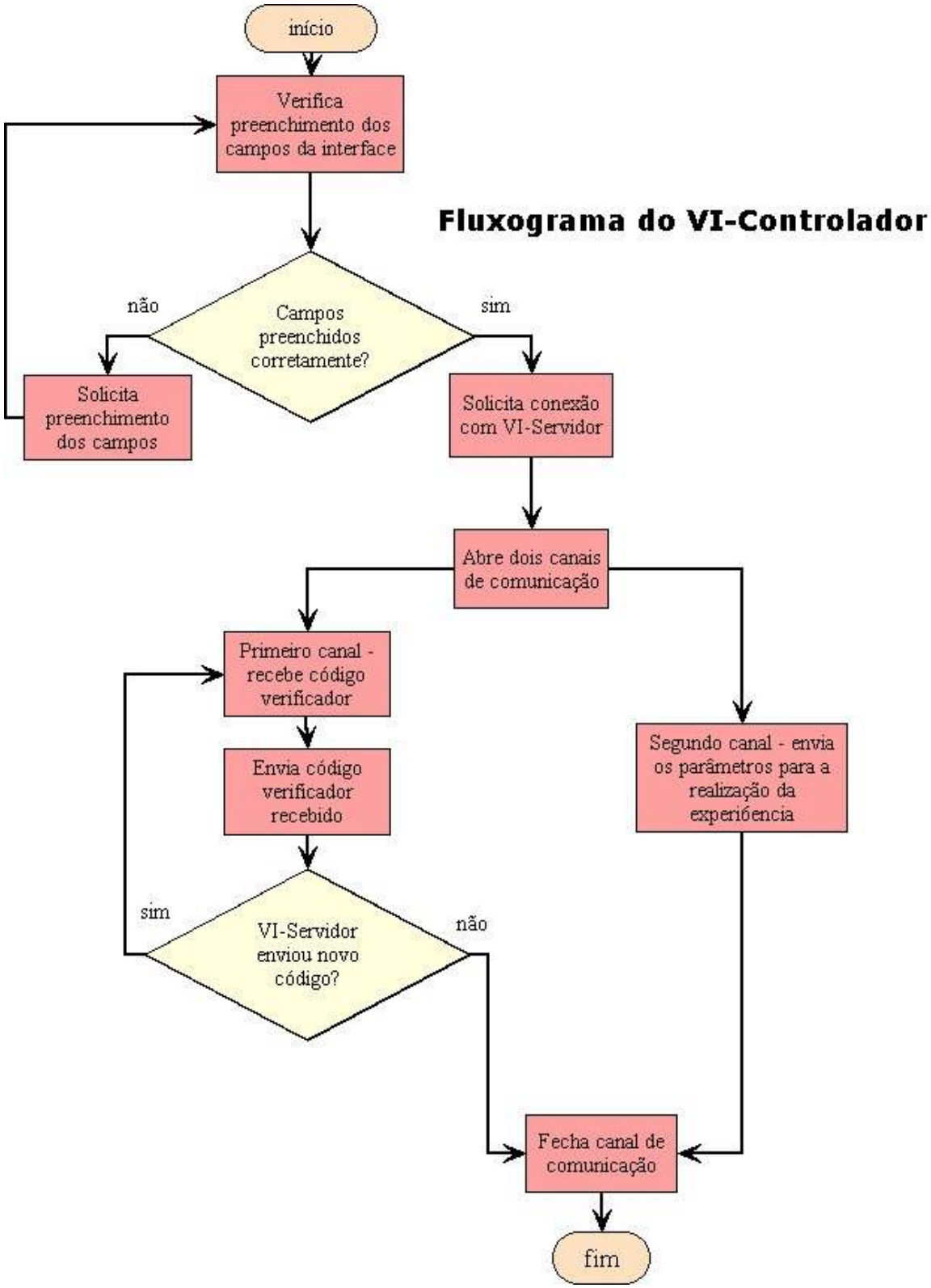

Figura 20 - Fluxograma do programa VI-Controlador.

4.2.3 O VI-Visualizador

O último programa desta fase de implementação do sistema, denominado **VI-Visualizador** cuja interface está ilustrada na **Figura 21**, é o responsável por mostrar os valores recebidos, adquiridos do aparato experimental. São indicados os parâmetros enviados pelo cliente (**VI-Controlador**), o nome do microcomputador onde está localizado o VI-Servidor bem como os parâmetros recebidos.

Figura 21 - Interface do VI-Visualizador implementado em *LabVIEW* responsável por mostrar os dados adquiridos da experiência.

O VI-Visualizador é o responsável pela visualização da experiência, ou seja, ele exibe os valores resultantes recebidos do **VI-Servidor**. De acordo com a Interface é possível visualizar o que se passa no servidor, não importando se a experiência foi ou não solicitada pelo usuário em questão.

Ao iniciar o programa ele cria um canal de comunicação com o servidor para verificar periodicamente a existência de controle da experiência. Caso afirmativo, o

programa passa a receber os dados da experiência até que esta esteja concluída; caso contrário é dado início ao processo de cancelamento.

Os campos da interface que merecem destaque são:

- **Intervalo de tempo entre amostras:** indica por quanto tempo serão amostrados os canais para uma mesma temperatura.
- **Tensão Inicial:** Indica com qual voltagem o programa iniciará a experiência.
- **Tensão Final:** Indica a última voltagem a ser imposta aos aquecedores.
- **Tensão de Incremento:** indica qual o incremento de voltagem após o termino do tempo de amostra.
- **Server name:** Nome do servidor onde está a experiência.
- **Status canal:** Indica qual é a situação do canal indicado.
- Canal 0 é a leitura do diodo; o Canal 1 é a leitura do termistor e o canal 2 é a leitura do aquecedor e Temperatura é a leitura direta do termopar.

O fluxograma que descreve as tarefas executadas pelo **VI-Visualizador** é apresentado na **Figura 22**.

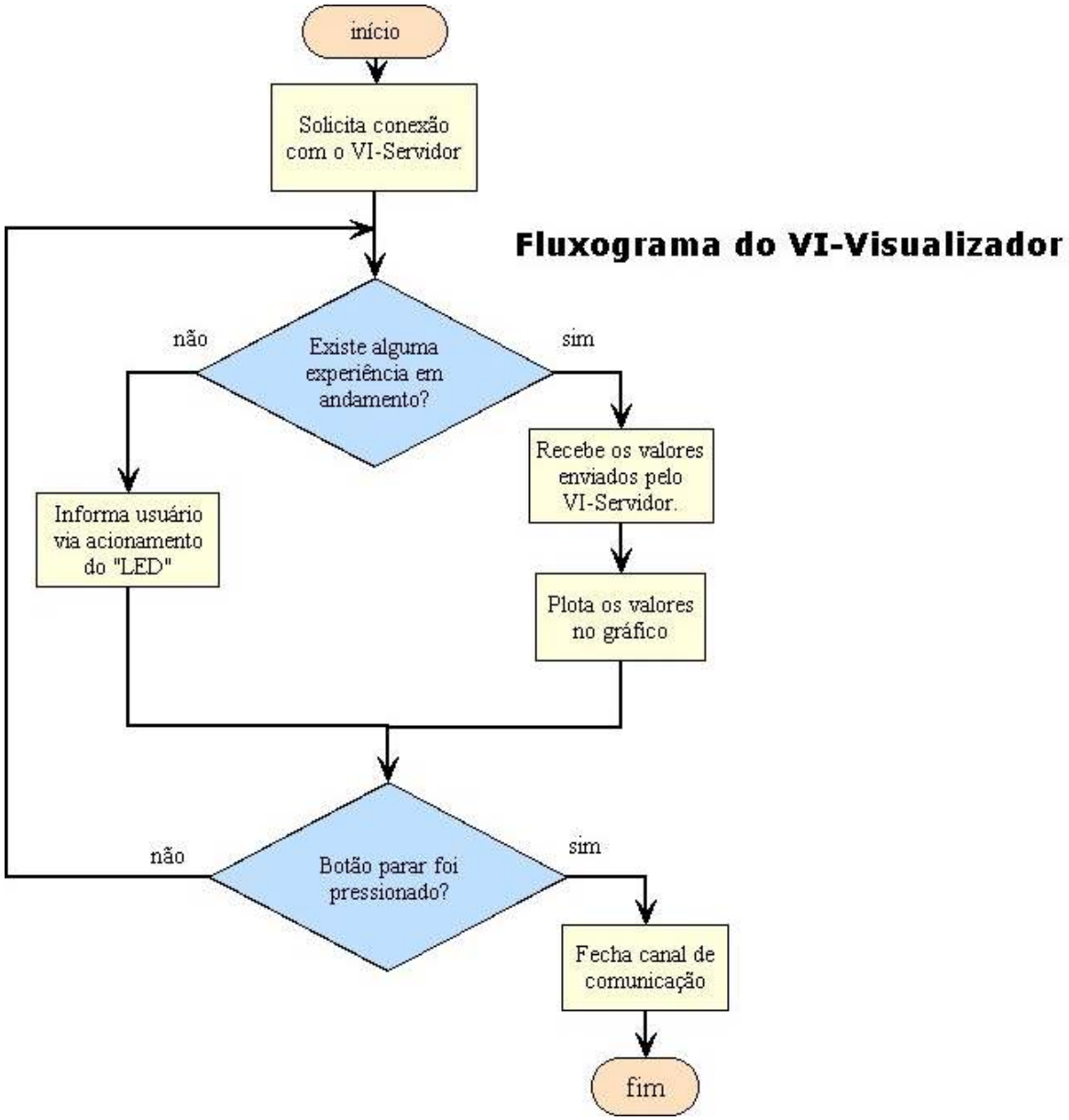

Figura 22 - Fluxograma do VI-Visualizador.

4.2.4 Conclusões Parciais

Nesta fase foi implementado um instrumento virtual (VI), denominado Aquisição SCXI para tornar possível, através da instrumentação virtual, a leitura dos valores de

tensão nos componentes Diodo, Termistor e Resistor, além da leitura de um termopar do tipo J possibilitando a caracterização dos componentes contidos no aparato experimental.

É possível verificar no diagrama de blocos apresentado as funções utilizadas para leitura de tensão nos canais 0, 1, 2 e temperatura onde o canal 0 é o responsável pela leitura da tensão no Diodo; o canal 1 é o responsável pela leitura da tensão no termistor, o canal 2 corresponde a leitura da tensão no resistor e finalmente o canal temperatura fornece a temperatura lida do termopar.

Este VI denominado Aquisição SCXI será inserido como um subVI onde a entrada deverá ser um valor de tensão e a saída deverá ser os valores lidos dos componentes. Esta será a segunda etapa a ser executada.

Foram implementados e testados três programas (VI-Servidor, VI-Controlador e VI-Visualizador) que compõem o sistema de controle remoto baseado em *LabVIEW®*. O resultado, apesar de satisfatório ainda não atendia a necessidade em distribuir um programa controlador sem a necessidade do software proprietário, no caso*, LabVIEW®*. Para atingir este objetivo a próxima etapa de desenvolvimento envolve a implementação de um programa que seja executado dentro de um *browser Web*, que seja mais amigável e menos trabalhoso para o usuário. No estado atual o usuário deve fazer o *download* e instalar mais de um arquivo para poder controlar o aparato experimental.

4.3 Segunda Etapa – Sistema Cliente/Servidor *LabVIEW® - ActiveX*

Nesta etapa foi elaborado um sistema de comunicação cliente/servidor para tornar possível o controle de instrumentos de forma remota via Internet, fazendo uso de um *Web browser* comercial.

Este sistema é composto por:

- Um software denominado LabControl - que é o cliente, implementado em *Visual Basic*. A partir deste cliente foi possível implementar e inserir o controle do

experimento em uma página *Web* utilizando a tecnologia *ActiveX*, disponível no ambiente de programação utilizado.

- Um software denominado **ServidorVB** - que funciona como servidor e mantém a comunicação entre o cliente (**LabControl**) via protocolo TCP/IP e com o instrumento virtual via protocolo DSTP.
- Um software servidor do experimento, **VI-Servidor**, que trata da instrumentação virtual.
- Um software denominado *DataSocket Server®*– toolkit desenvolvido pela *National Instruments* para tornar possível a comunicação entre equipamentos de medidas e softwares aplicativos que enviam comandos para estes equipamentos via Internet.

O programa elaborado em *Visual Basic*, denominado **LabControl** cuja interface é apresentada na **Figura 22**, contém os campos existentes no **VI-Controlador** e no **VI-Visualizador**, ou seja, o usuário pode solicitar o controle do experimento através do botão Calcular, após ter preenchido os valores de tensão inicial, tensão final, tempo de amostragem e tensão de incremento; e escolhido qual valor será exibido no gráfico. Através do **LabControl** é possível enviar os parâmetros necessários para fazer as medidas de temperatura de um termopar, diodo, termistor e resistor, ou seja, do aparato experimental, além de receber os dados vindos do **ServidorVB**, resultantes da experiência realizada. O aparato experimental está fisicamente localizado no Laboratório de Microeletrônica (LME) na Escola Politécnica (POLI) da Universidade de São Paulo (USP),

O **ServidorVB** está instalado em um microcomputador pessoal cujo processador é um K6-II com 450MHz, localizado próximo ao aparato experimental, também no LME-POLI-USP. É responsabilidade do **ServidorVB** manter a comunicação com o **LabControl**, receber os parâmetros vindos do **LabControl**, repassá-los ao *DataSocket Server®*, receber a resposta do *DataSocket Server®* e enviar esta resposta ao **LabControl**, para que seja mostrada graficamente ao usuário.

A **Figura 23** mostra a interface do **LabControl**, software cliente que tem como função enviar e receber parâmetros via Internet e exibir valores na forma gráfica. Nesta

ilustração é possível verificar o software em seu estado inicial, ou seja, apenas com os valores enviados para o **VI-Servidor**. O botão Calcular é modificado para Aguarde Processando... para que o usuário possa perceber que a comunicação com o servidor foi estabelecida e que os dados estão sendo enviados.

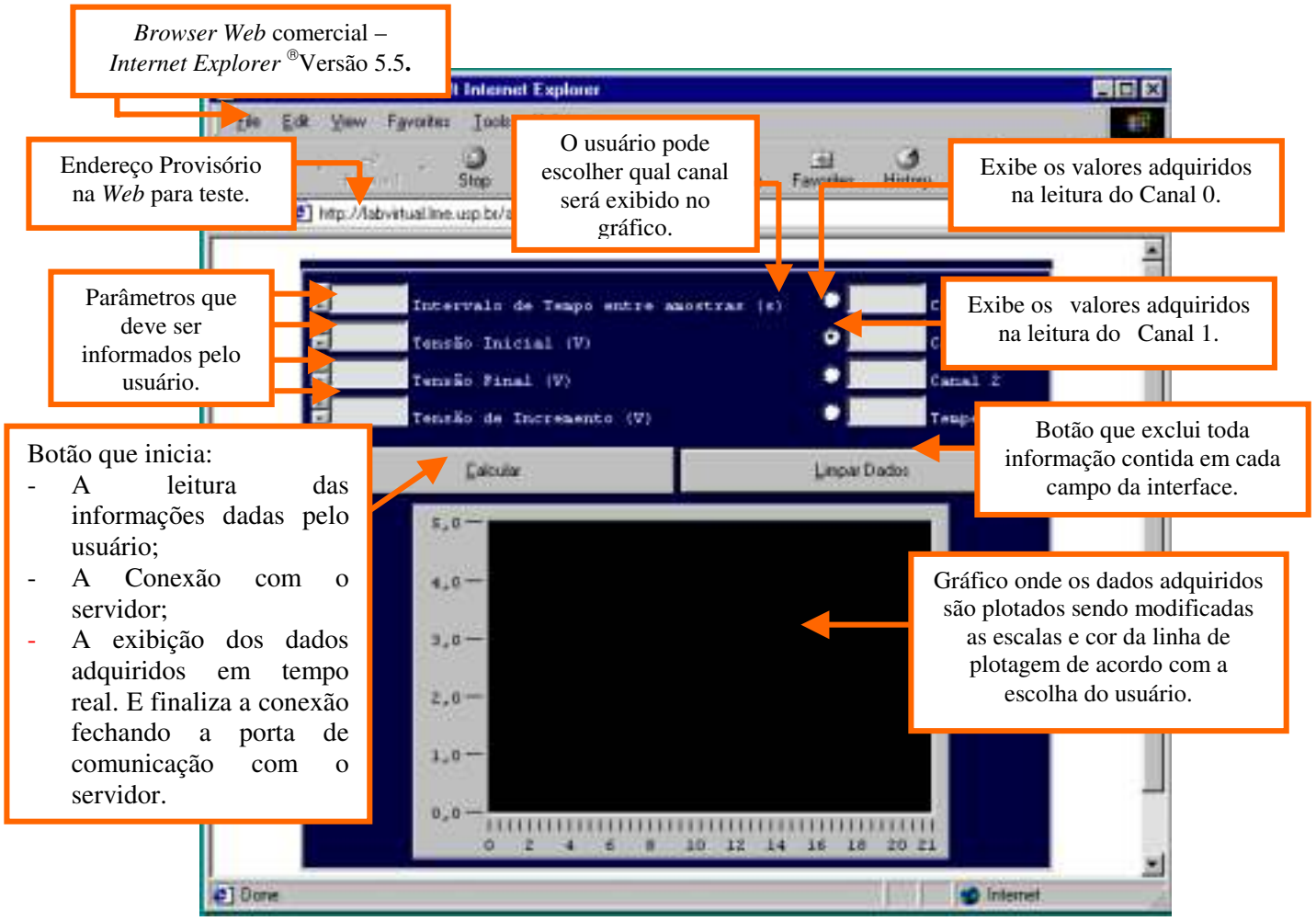

Figura 23 - Interface do aplicativo LabControl embutido em um *browser Web* comercial possibilitando o acesso remoto.

A **Figura 24** ilustra a arquitetura de comunicação cliente/servidor implementada nesta etapa de desenvolvimento.

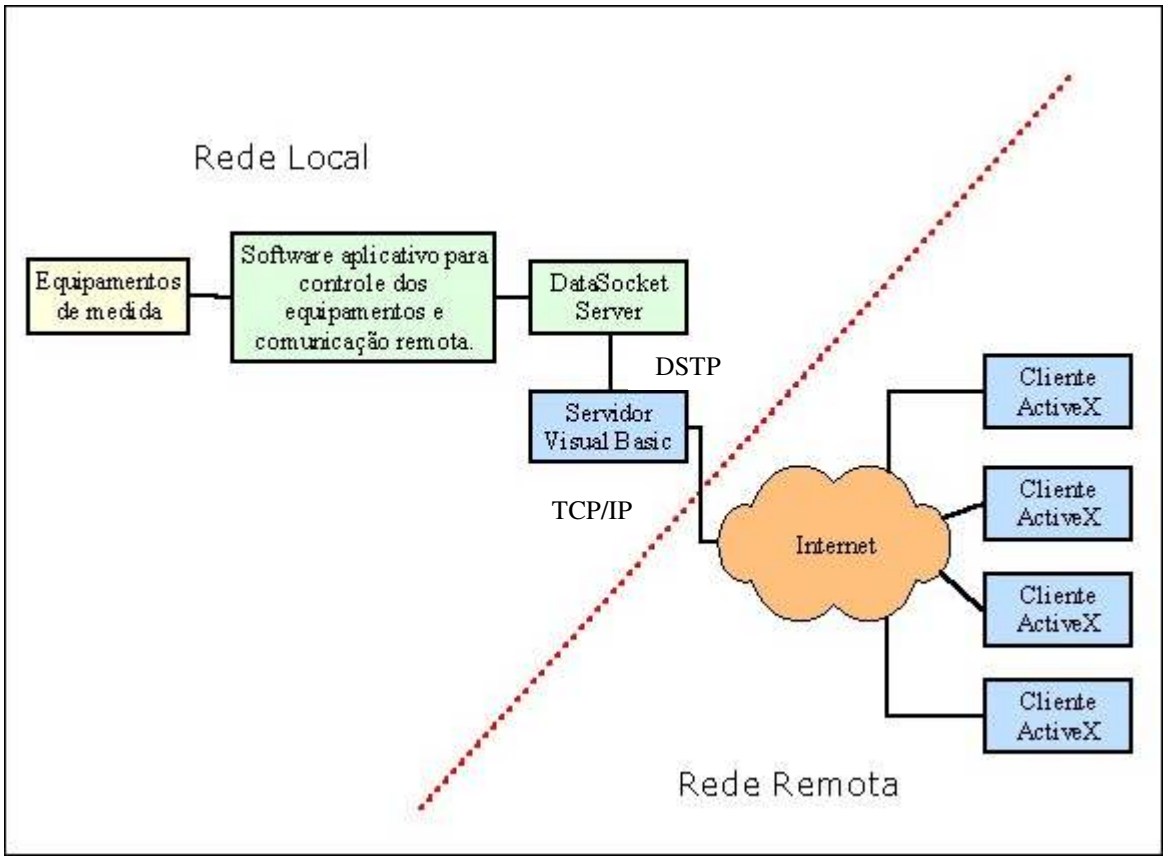

Figura 24 - Arquitetura da comunicação cliente/servidor entre aplicativo VI-Servidor e aplicativos LabControl embutido em browser *Web*.

O **ServidorVB**, cuja interface está mostrada na **Figura 25**, foi implementado para gerenciar os acessos dos usuários, sendo responsável por:

- Criar uma fila de clientes interessados em controlar o experimento;
- Gerenciar a fila de acessos dos clientes;
- Receber dados do cliente e enviá-los ao *DataSocket Server®*. Como já mencionado anteriormente o *DataSocket Server®* é um servidor comercial desenvolvido pela *National Instruments*.
- Receber os dados enviados pelo *DataSocket Server®* via protocolo *DSTP* e enviá-los ao cliente solicitante e aos outros clientes conectados.
- Identificar o número IP do usuário conectado.
- Identificar a porta de comunicação utilizada pelo ultimo usuário conectado.

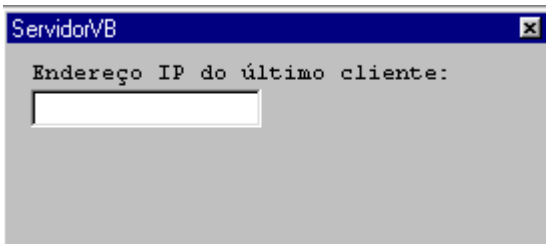

Figura 25 - Interface do programa ServidorVB.

A interface do **LabControl** mostrada na **Figura 26** ilustra o recebimento dos dados onde é possível verificar os valores recebidos no campo superior direito bem como o gráfico que vai sendo mostrado conforme são recebidos os valores via protocolo *TCP/IP*.

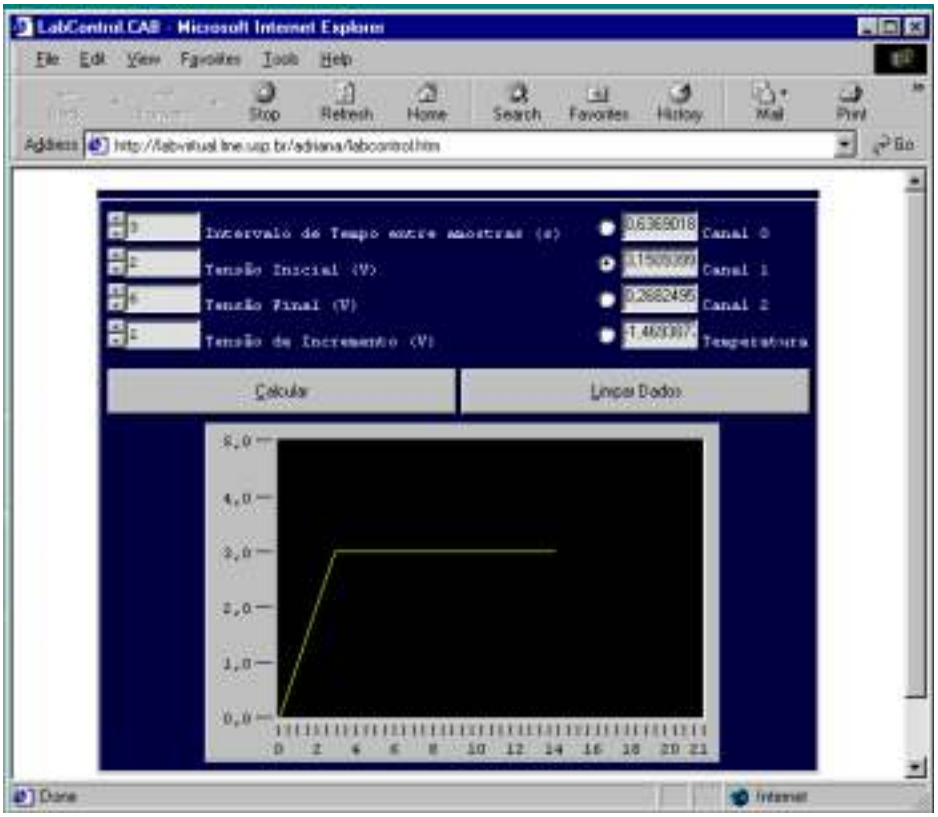

Figura 26 - Interface do LabControl onde é possível verifica a recepção dos dados nos campos localizados no canto direito bem como o gráfico escolhido pelo usuário.

Para que os dados pudessem ser analisados de uma forma mais adequada o modo de plotagem do gráfico foi alterado, dando origem à interface mostrada na **Figura 27**, ou seja, os valores dos quatro canais são exibidos no mesmo gráfico, sem que o usuário necessite escolher uma curva a ser desenhada. Este novo plotador é um programa *ActiveX*, embutido no **LabControl**. Também foram realizados alguns ajustes na aquisição de dados em uma tentativa de corrigir os valores incorretos que eram lidos.

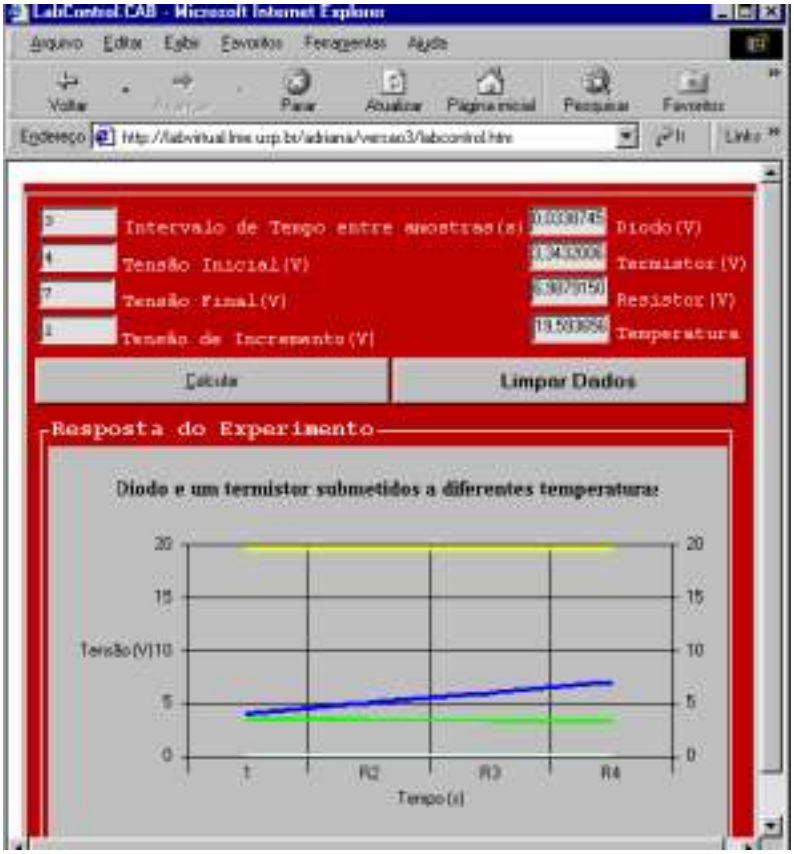

Figura 27 - Nova interface do LabControl onde é possível verificar a recepção dos dados bem como a plotagem dos quatro valores recebidos no gráfico.

4.3.1 Conclusões parciais

Nesta etapa foi possível ajustar a leitura de alguns parâmetros que estavam incorretos alterando canais de leitura a partir do diagrama de blocos do programa VI-Servidor do *LabVIEW®*.

Foram implementadas, em *Visual Basic*, duas versões do controle para ser inserido em um *browser Web*. Na primeira versão havia a possibilidade de escolher qual dos canais teria seus valores mostrados no gráfico. Analisando o funcionamento do programa optou-se por mostrar todas as curvas em um mesmo gráfico, o que facilitaria a visualização dos dados obtidos.

Foi possível estabelecer a comunicação entre dois aplicativos desenvolvidos em linguagens de programação diferentes (*LabVIEW®* e *Visual Basic*) fazendo com que um deles, o **LabControl,** enviasse parâmetros numéricos ao outro, **VI-Servidor**, via Internet através do protocolo proprietário DSTP e também do protocolo da Internet, TC/IP.

Nos testes de acesso remoto através de linha discada, observou-se um atraso na transmissão de dados além de incompatibilidade entre o protocolo utilizado pelo **VI-Servidor** (DSTP – *DataSocket Transfer protocol*) e outros programas que utilizam o protocolo padrão da Internet (TCP/IP). Esta incompatibilidade causava um travamento da máquina impossibilitando o funcionamento do Laboratório Virtual por um longo período de tempo (maior que seis horas ao dia).

Pela necessidade pontual em obter o máximo de utilização de um mesmo microcomputador onde deve estar funcionando mais de um programa com conexão via Internet, optou-se por desenvolver um novo cliente abandonando o existente desenvolvido em *Visual Basic* com a tecnologia *ActiveX*. Desta forma foi definida mais uma etapa a ser implementada para suprir a necessidade de desenvolver um cliente que se comunicasse com um servidor via TCP/IP, o que possibilitaria a execução de um número maior de aplicativos ao mesmo tempo na mesma máquina servidora. Esta modificação deve oferecer maior flexibilidade de acesso para que o programa cliente possa ser executado em qualquer sistema operacional ou *browser Web*.

4.4 Terceira Etapa – novo sistema Cliente/Servidor

O objetivo desta fase é otimizar os programas implementados tornando-os mais flexíveis e portáveis, ou seja, implementando um sistema que funcione

independentemente do tipo de sistema operacional ou do *browser Web*. Para isso foi preciso realizar algumas modificações no **VI-Servidor** bem como no cliente. Estas mudanças serão descritas nos tópicos seguintes.

4.4.1 Modificações no servidor

Devido a uma incompatibilidade do **VI-Servidor** com outros aplicativos desenvolvidos que funcionam na mesma máquina servidora o modo de comunicação precisou ser modificado. Nesta fase foram implementadas funções TCP/IP existentes no *LabVIEW®* para possibilitar a comunicação via rede de computadores.

Na nova interface do programa servidor, ilustrada na **Figura 28**, nesta fase denominado **Servidor_GSIM,** é possível identificar o endereço remoto, ou seja, o número IP da máquina do usuário que está controlando a experiência, o número da porta de conexão, os erros que acontecerem durante a execução do experimento ou comunicação com o cliente, a string de dados recebidos e a string de dados enviada. Este último campo vai sendo atualizado de acordo com a realização das medidas.

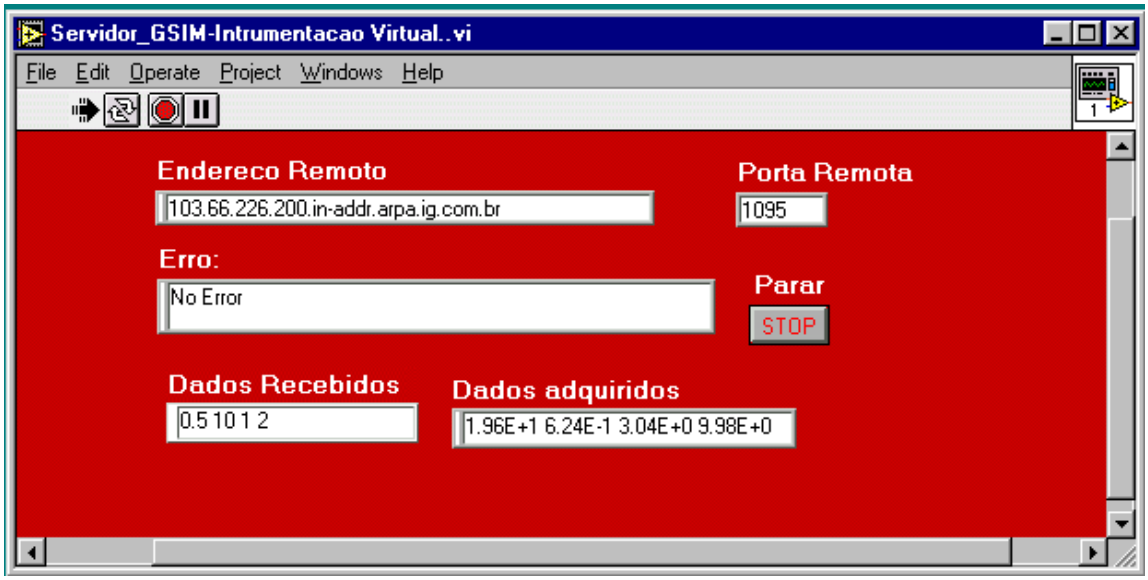

Figura 28 - Interface do programa servidor desenvolvido em *LabVIEW®* para estabelecer comunicação com o cliente remoto e adquirir os dados da experiência em tempo real.

Todo o processamento da informação no servidor fica dentro de um *loop* Enquanto (*While*). Todos os passos inseridos neste *loop* serão executados até que o botão PARAR seja pressionado. O programa possui ainda seis seqüências onde a primeira é a responsável por abrir um canal de comunicação e aguar a recepção de uma string de dados via protocolo *TCP/IP*. A primeira string possui tamanho de 6 bits e informa apenas o tamanho da próxima string que deve ser recepcionada pelo *TCP/IP*.

Todos os dados chegam como string e são transformados em dados numéricos, pois as funções do *TCP/IP* existentes no *LabVIEW®* só manipulam dados numéricos. Além disso, os dados chegam como string, pois a linguagem de programação Java, com a qual o novo cliente foi desenvolvido, só permite que sejam enviados e recebidos dados do tipo string. Esta seqüência é ilustrada pela **Figura 29**.

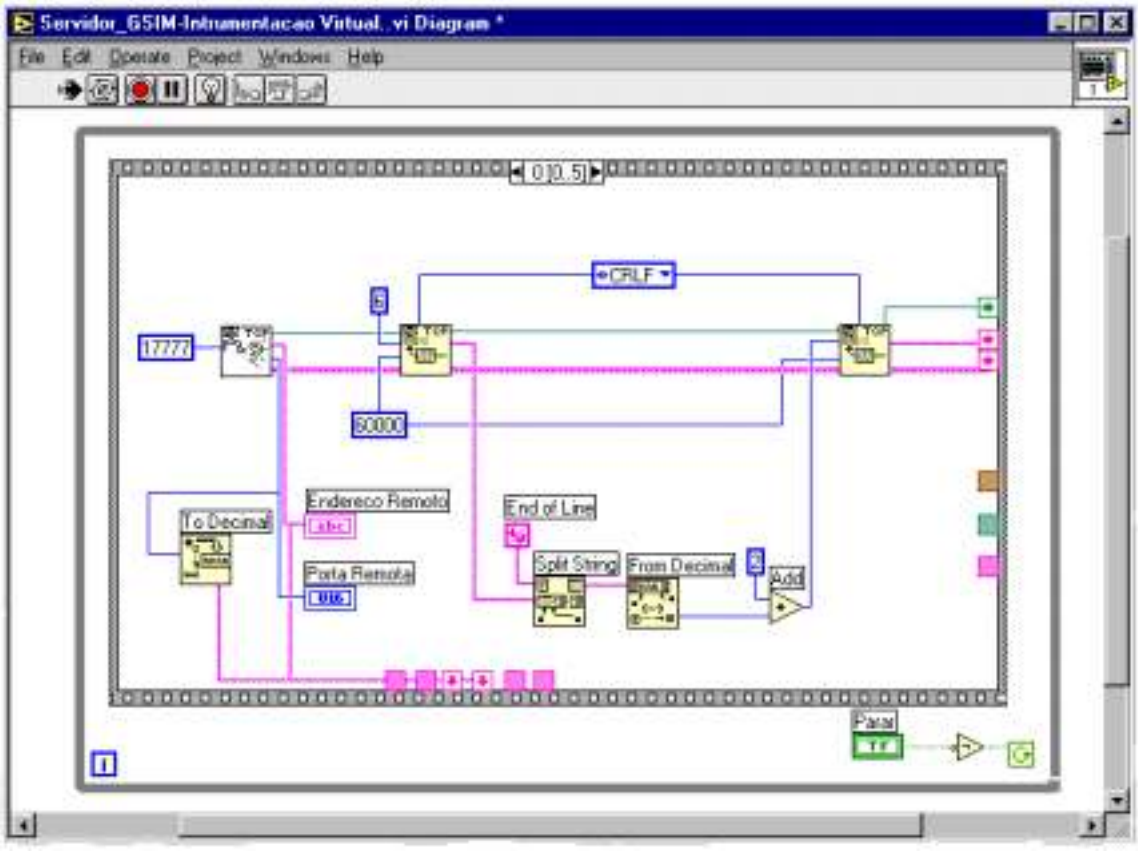

Figura 29 - Diagrama de blocos desenvolvido no *LabVIEW®* onde podem ser visualizados o *loop* e a seqüência 0 do programa Servidor_GSIM.

Na segunda seqüência, ilustrada na **Figura 30,** ocorre apenas obtenção da data e hora atual. Estes dados serão utilizados pela última seqüência.

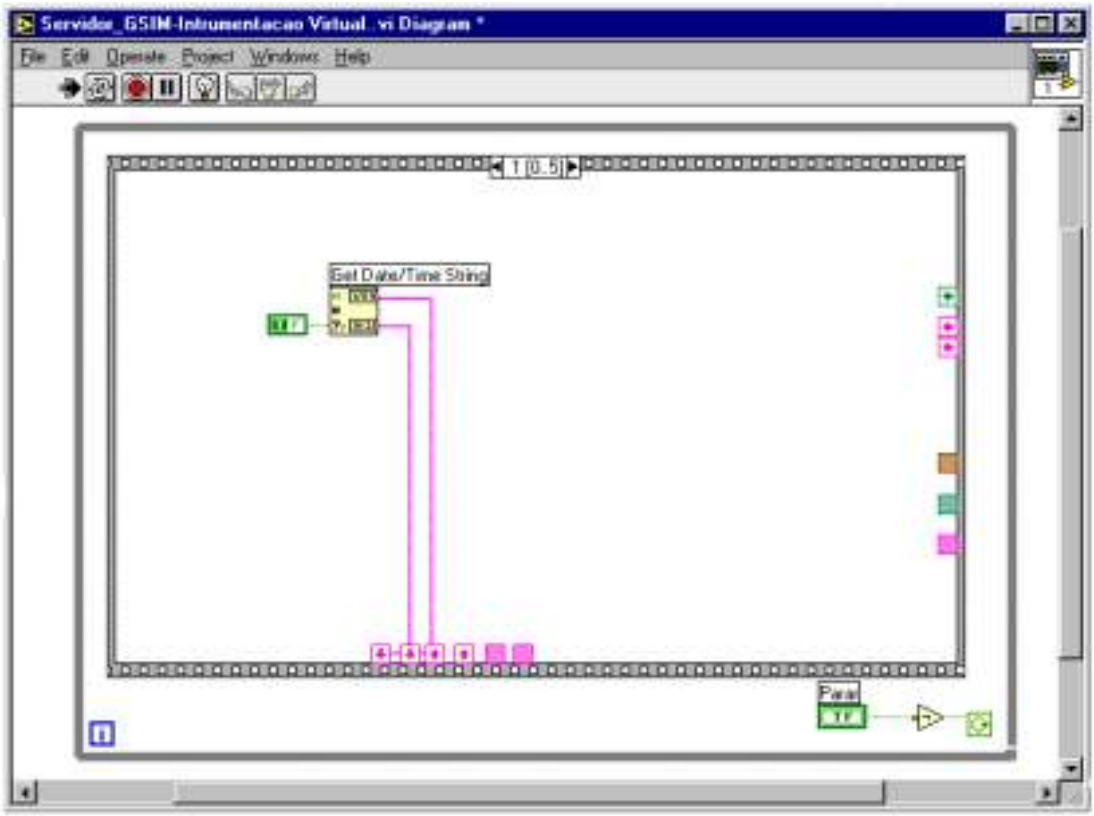

Figura 30 - Diagrama de blocos ilustrando a transformação dos dados string em dados numéricos.

A terceira seqüência, ilustrada pela **Figura 31**, é responsável pela transformação dos dados de string para numéricos para que a aquisição possa ser iniciada.

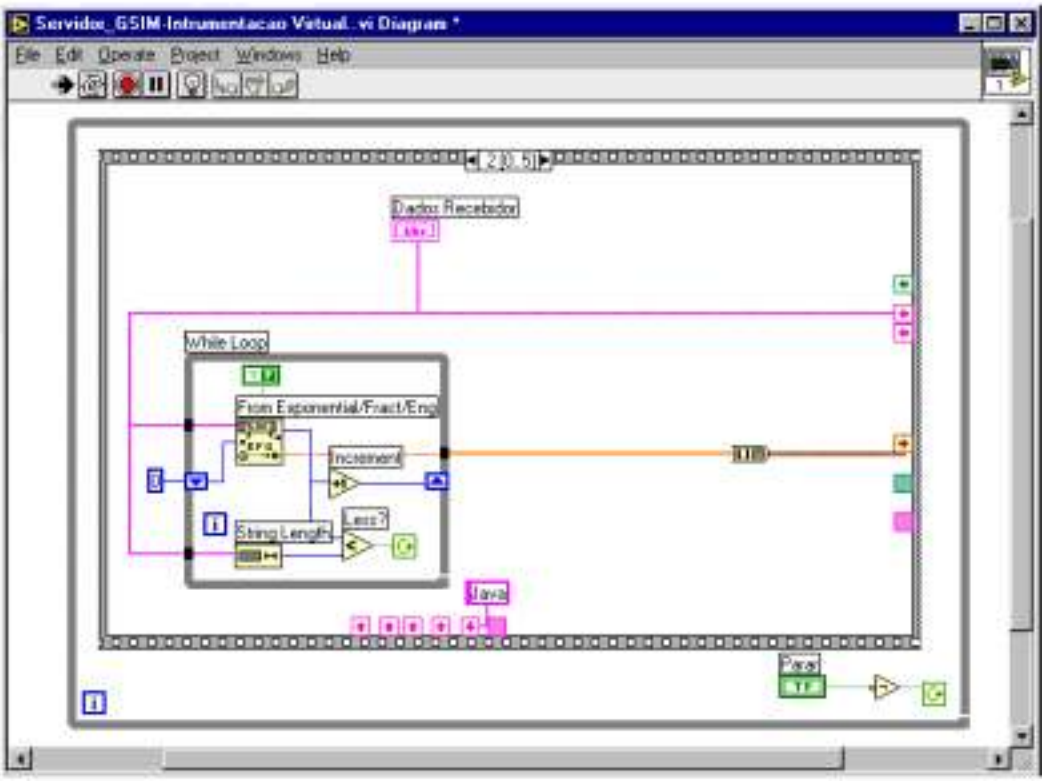

Figura 31 - Diagrama de blocos que ilustra a seqüência mais importante onde existe uma chamada a um SubVI responsável pela aquisição dos dados.

Na seqüência 3 (**Figura 32**) obtêm-se os valores resultantes da amostragem, os quais devem ser transformados de numérico para string e enviados ao cliente via protocolo *TCP/IP*. Esta é a seqüência mais importante pois é onde acontece a aquisição dos dados. Existe a chamada a um SubVI (Aquisição SCXI) responsável pela leitura dos valores de tensão dos componentes envolvidos na experiência. O valor da tensão inicial, já transformada de string para numérico na seqüência anterior, é o parâmetro de entrada do SubVI responsável pela aquisição dos dados.

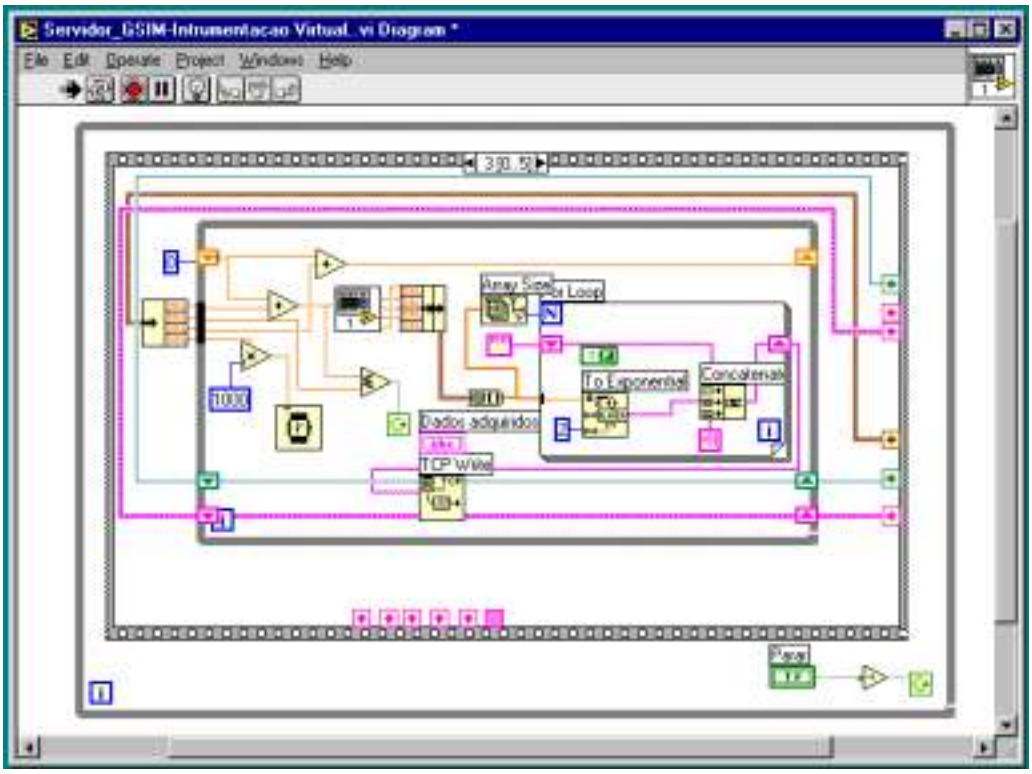

Figura 32 - Diagrama de blocos ilustrando o envio do pacote de dados realizado pela função TCPwrite do próprio *LabVIEW®*.

O envio dos dados ocorre na penúltima seqüência do programa, onde as variáveis de entrada recebem valor zero e o programa aguarda até que novos dados sejam enviados (**Figura 33**) pelo cliente via Internet.

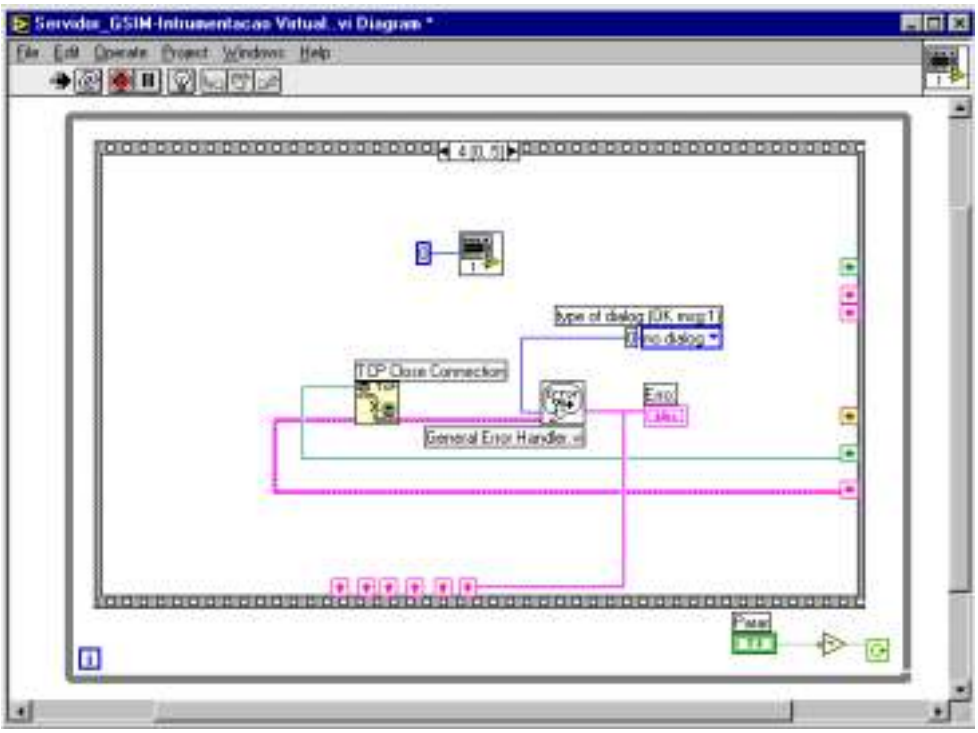

Figura 33 - Quarta seqüência responsável pelo envio dos dados para o cliente solicitante.

No fim do processo algumas informações são armazenadas no microcomputador servidor onde o programa descrito funciona. A quinta seqüência (**Figura 34**) é a responsável por criar o arquivo que armazena as informações sobre o tempo de acesso (horário inicial e horário final); sobre o número IP do microcomputador que solicitou o controle; a data atual e o erro, caso tenha ocorrido algum durante o processo. Este procedimento foi implementado para auxiliar na monitoração de funcionamento do servidor e no tempo de resposta do experimento.

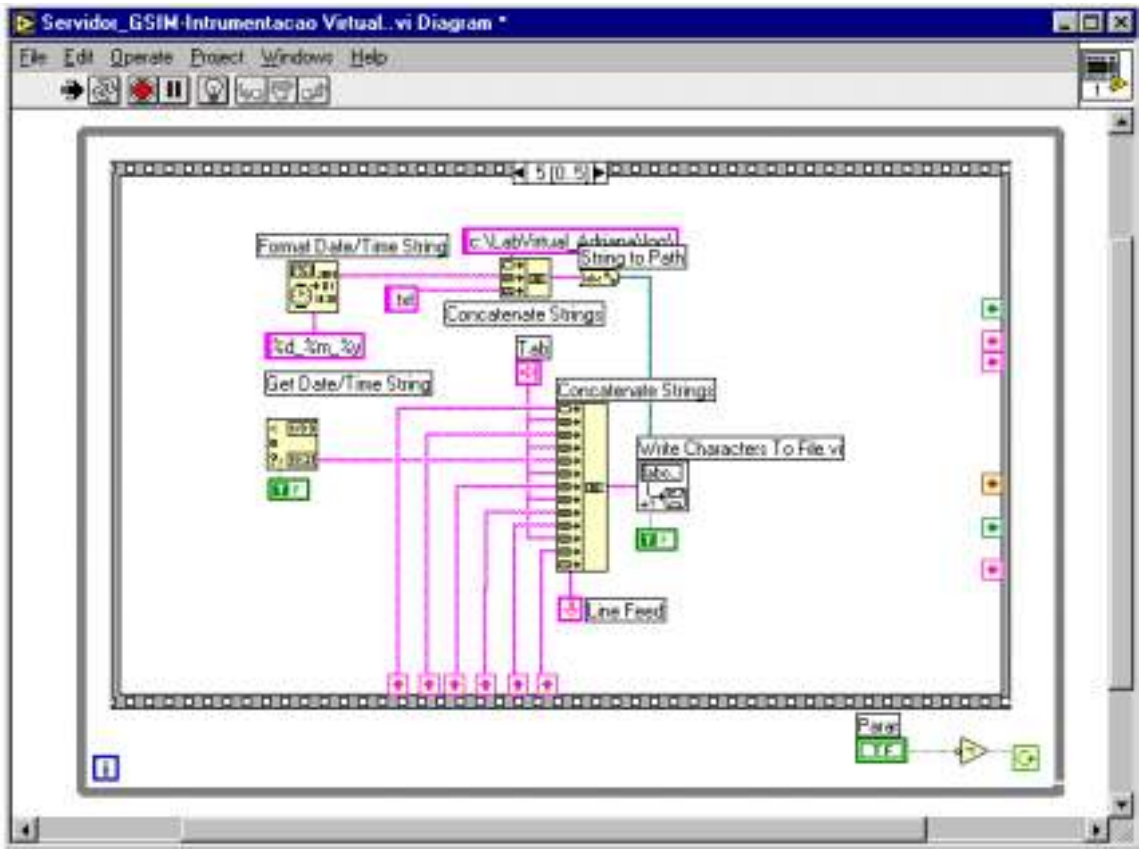

Figura 34 - Seqüência onde é criando um arquivo com informações sobre a máquina do usuário e o tempo de resposta do aparato experimental.

O princípio de funcionamento do programa servidor desenvolvido que faz uso das funções TCP/IP disponíveis no *LabVIEW*® é ilustrado pela **Figura 35**.

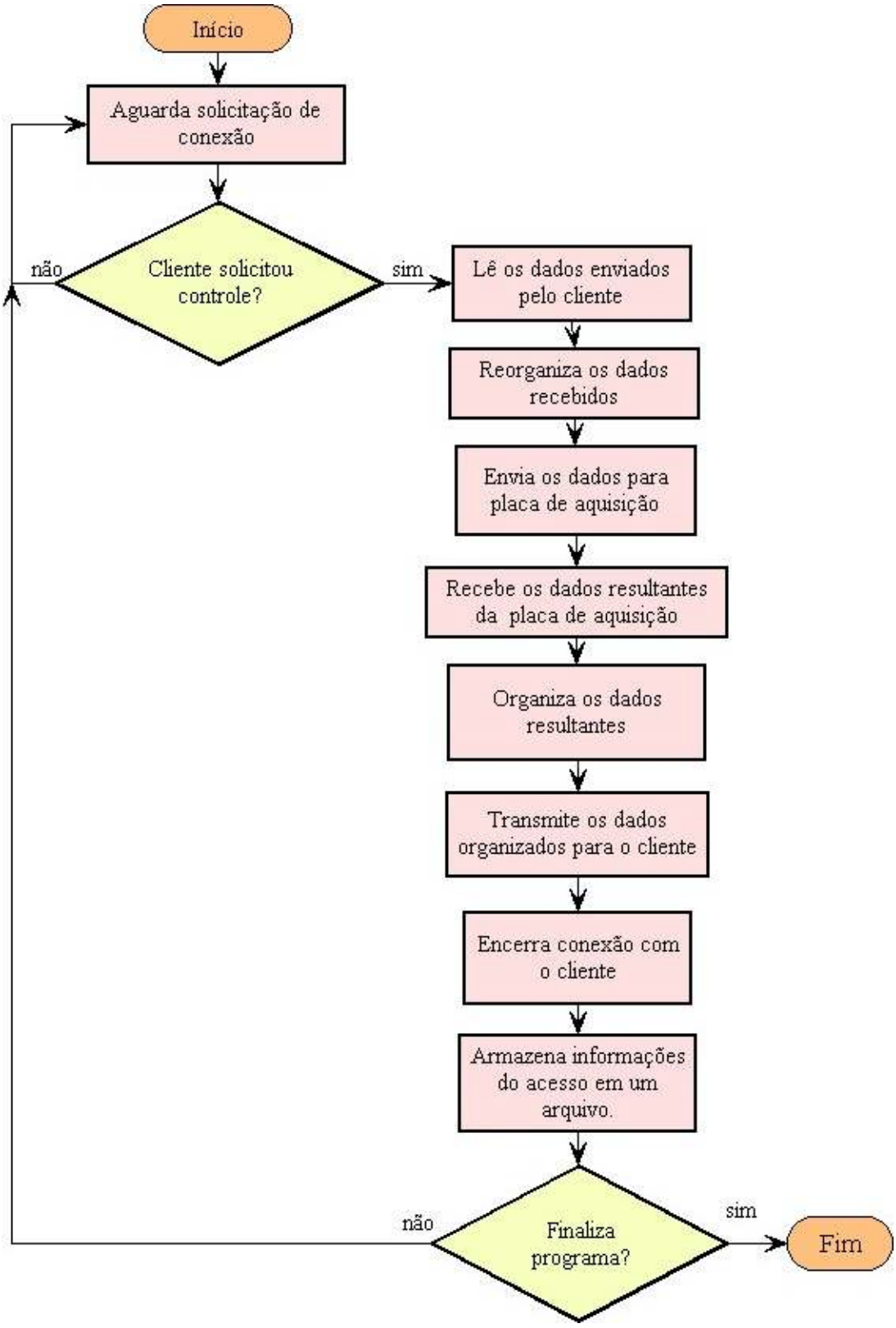

Figura 35 - Fluxograma do Programa servidor desenvolvido em *LabVIEW*®.

4.4.2 Modificações no cliente

Para implementar um aplicativo cliente mais flexível para estabelecer a comunicação com o **Servidor_GSIM** desenvolvido em *LabVIEW®* optou-se pela modificação da linguagem de programação. Sendo assim, nesta fase o cliente deve ser desenvolvimento em linguagem *Java*, possibilitando que o aplicativo seja executado em qualquer *browser Web*.

A interface desenvolvida para ser executada embutida em um *browser Web* está mostrada na **Figura 36**. Nesta interface o usuário digita os valores da tensão inicial, da tensão final, da tensão de incremento e do tempo de amostragem, da mesma forma realizada na interface anterior desenvolvida em *Visual Basic*. O campo Status exibe o que está acontecendo durante a execução da experiência. O botão Enviar Dados é responsável por chamar a função que estabelece a comunicação com o **Servidor_GSIM** através do protocolo TCP/IP. Os dados recebidos são exibidos nos gráficos. O botão RESET limpa os campos, permitindo que os parâmetros da experiência sejam digitados novamente.

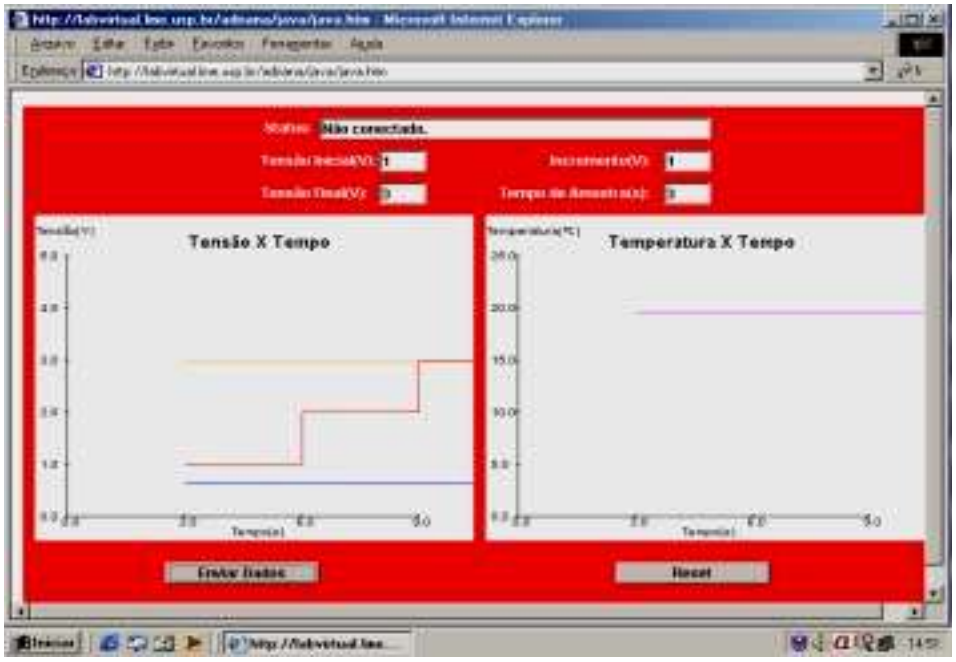

Figura 36 - Interface elaborada com a linguagem Java para funcionar embutida em um *browser Web* comercial.

Com relação às linhas exibidas no gráfico 1, a laranja corresponde aos valores da tensão lida do termistor; a azul indica os valores da tensão lida do diodo e a vermelha indica os valores da tensão lida do resistor (aquecedor). A **Tabela 1** apresenta as funções de cada objeto presente na interface do programa.

Interface	Função:
Status	Informa ao usuário o estado atual do funcionamento do programa, como, por exemplo, enviando ou recebendo dados.
Tensão Inicial	Campo onde o usuário deverá informar o valor de tensão inicial em volts, onde o valor máximo aceitável é nove volts.
Tensão Final	Campo onde o usuário deverá informar o valor de tensão final em volts, onde o valor máximo aceitável é dez volts.
Tensão de Incremento	Campo onde o usuário deverá informar o valor da tensão de incremento para a realização da experiência, onde o valor máximo aceitável é um volt.
Tempo de amostragem	Campo onde o usuário deverá informar o tempo de amostragem, ou seja, o tempo de espera entre as medidas.
Gráfico 1	Os dados referentes à tensão são apresentados no primeiro gráfico, Tensão X Tempo.
Gráfico 2	Os valores de temperatura são exibidos no segundo gráfico, Temperatura X Tempo.
Botão Enviar	Envia ao programa **Servidor_GSIM**, via Internet, os valores informados pelo usuário para que a experiência tenha início.
Botão Reset	Limpa a tela, eliminando os valores dos campos e dos gráficos.

Tabela 1 – Função de cada objeto presente na Interface do programa cliente.

Este aplicativo é um *applet* (pequeno programa em java) que se auto-instala no microcomputador do usuário. Por questões de segurança e por se tratar de aplicativo executado em poucos segundos, os dados adquiridos não são salvos. Caso ocorra algum erro o servidor enviará uma mensagem informando ao usuário o acontecido. Esta mensagem aparecerá no campo Status.

Nesta fase o princípio de funcionamento do programa cliente não sofreu alterações. Apenas a interface e a linguagem de programação foram modificadas. A

Figura 37 ilustra o princípio de funcionamento dos programas cliente implementados em *ActiveX* (na primeira fase) e em *Java* (na Segunda fase de desenvolvimento).

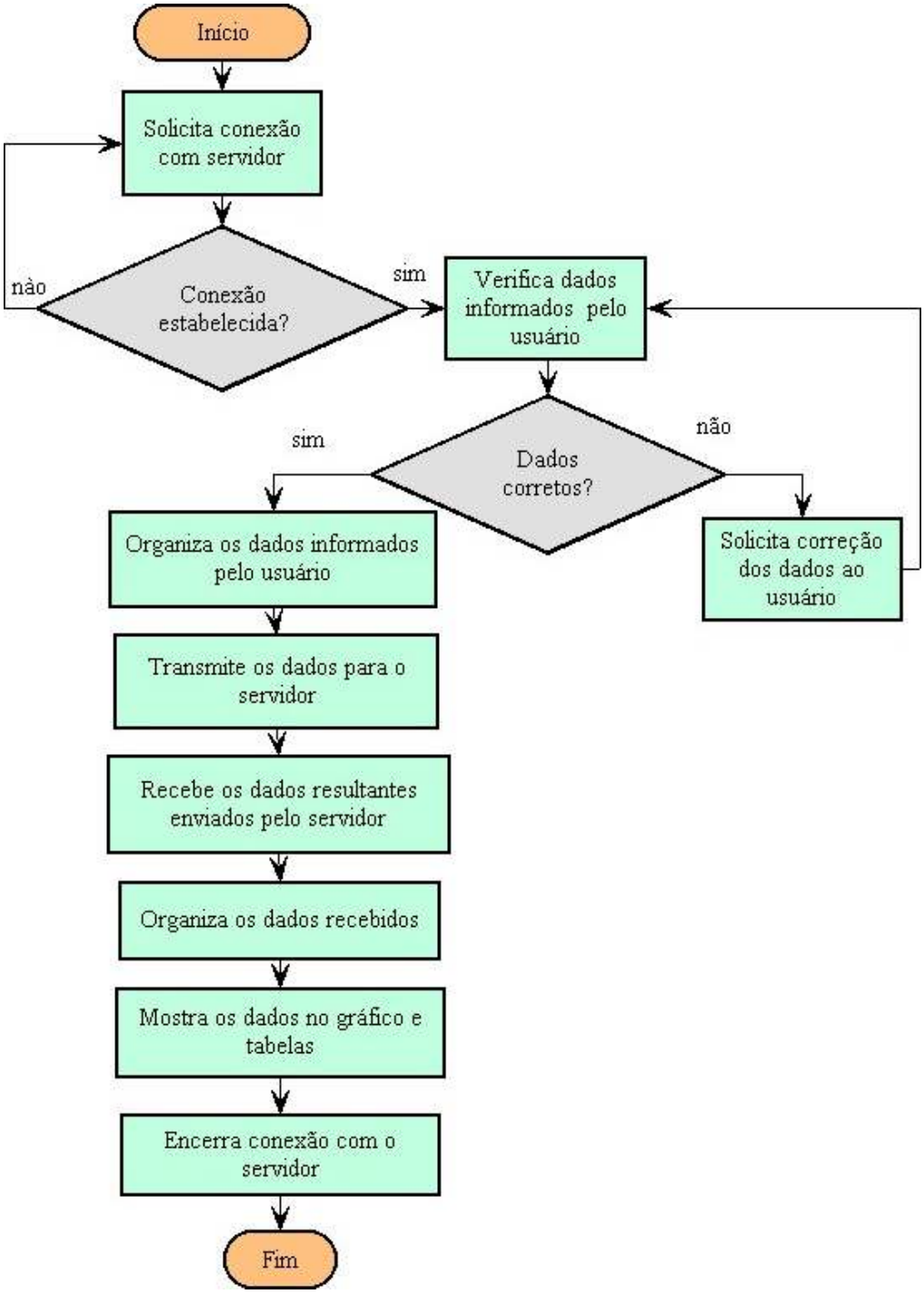

Figura 37 - Fluxograma do Programa Cliente desenvolvido em Java e *ActiveX*.

4.4.3 Tempos das Transmissões dos dados

Foram monitoradas seis rotas, distribuídas nos continentes ilustrados pela **Figura 38,** para comprovar a viabilidade de uso do laboratório virtual, sendo este completamente dependente do funcionamento de redes de computadores sejam elas locais e/ou de longas distâncias.

Figura 38 = Mapa do mundo com destaque aos países cujas rotas foram monitoradas.

Os endereços na Internet, mostrados na **Tabela 2,** foram selecionados com a restrição de pertencerem a Universidades distribuídas na maior extensão geográfica possível. Estes endereços, também chamados de rotas, foram monitorados durante quinze dias com o auxílio do software denominado *PingPlotter* [LINK 15]. A cada quinze minutos eram enviados pacotes de tamanho 25 KB contendo os programas necessários para estabelecer a comunicação com o **Servidor_GSIM** (applet1.java, plotador1.java, plotador2.java e java.htm) e ter acesso ao controle do experimento desenvolvido neste trabalho.

Continente – País	Endereço – Universidade
América do Sul – Chile São Paulo	http://www.dcc.uchile.cl – Universidade do Chile http://www.unicamp.br – Universidade Estadual de Campinas
América do Norte – Estados Unidos	http://www.stanford.edu – Universidade de Stanford
Europa – Alemanha	http://www.uni-bonn.de – Universidade de Bonn
África – Israel	http://www.bgu.ac.il – Universidade Ben Gurion de Negev - Israel
Ásia – Japão	http://www.iuj.ac.jp - Universidade Internacional do Japão

Tabela 2– Endereços selecionados aleatoriamente divididos nos seis continentes.

Para avaliar o desempenho das rotas foram estabelecidos os seguintes critérios:

1. Tempo de resposta ótimo – até 0,10 segundos.
2. Tempo de resposta satisfatório – até 0,15 segundos.
3. Tempo de resposta ruim – maior que 0,2 segundos.

Este detalhamento é importante, pois mostra por quantos *hosts* (computadores, roteadores) a informação passou no percurso que liga o remetente, microcomputador utilizado como servidor do laboratório virtual, ao destinatário na rede de computadores.

O número de *hosts*, a média dos tempos de resposta e o percentual de pacotes perdidos durante a realização das medidas são mostrados na **Tabela 3.**

País	Número de *Hosts*	Média do Tempo de resposta	Pacotes perdidos
Alemanha	22	0,15274 segundos	10 %
Brasil	8	0,013 segundos	7,27 %
Chile	25	0,174 segundos	32,26 %
Estados Unidos	24	0,15253 segundos	38,84 %
Israel	18	0,149 segundos	15,99 %
Japão	21	0,232 segundos	14,61 %

Tabela 3 – Tempos médios de resposta e percentual de pacotes perdidos durante a realização das medidas no melhor e no pior caso.

A rota que apresentou o melhor desempenho foi a localizada no Brasil. Este resultado já era esperado por tratar-se de uma rede de menor extensão e menor número de *hosts* intermediários entre o microcomputador remetente e o destinatário. O caminho

que apresentou pior resultado, no que diz respeito à rota que apresentou maior tempo de resposta, foi o localizado no Japão, apesar de existirem 21 *hosts* e pouca perda de pacotes. A perda de pacotes indica que a informação deverá ser re-enviada. Esta é uma propriedade do protocolo TCP/IP, que permite que sejam verificados tanto o recebimento dos pacotes de dados como a estimativa do tempo de resposta do destinatário.

O gráfico apresentado na **Figura 39** demonstra a média do tempo de resposta que apresentou um resultado ótimo.

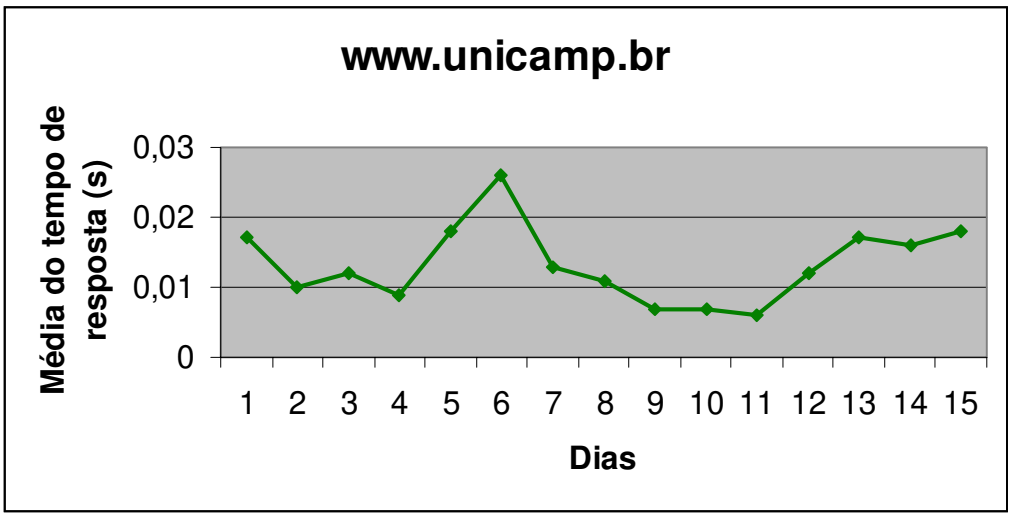

Figura 39 - Gráfico que demonstra a média do tempo de resposta para o envio de pacotes com 25 KB na rota que apresentou o melhor resultado.

Na **Figura 40** é apresentada a média do tempo de resposta do endereço que apresentou desempenho satisfatório. Incluindo o número de *hosts* e a porcentagem de pacotes perdidos, respectivamente 24 e 38,84%, é possível considerar ainda que esta rota, apesar do tráfego de informação, apresenta um desempenho considerável.

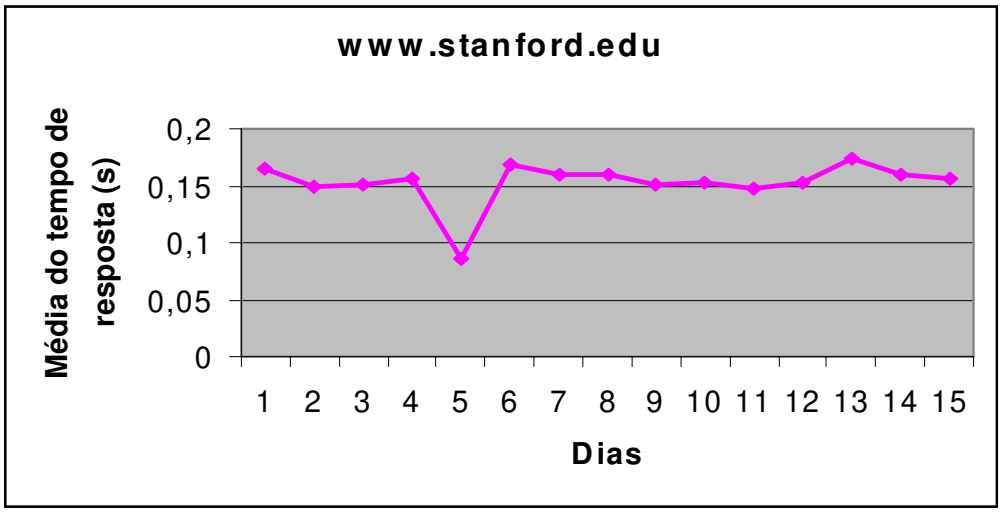

Figura 40 - Gráfico que demonstra a média do tempo de resposta para o envio de pacotes com 25 KB na rota que apresentou resultado intermediário.

Na **Figura 41** é apresentada a média do tempo de resposta do endereço que apresentou o pior desempenho, mesmo com o elevado número de *hosts* (vinte e um) interligando o remetente e o destinatário com apenas 14,61% de pacotes perdidos.

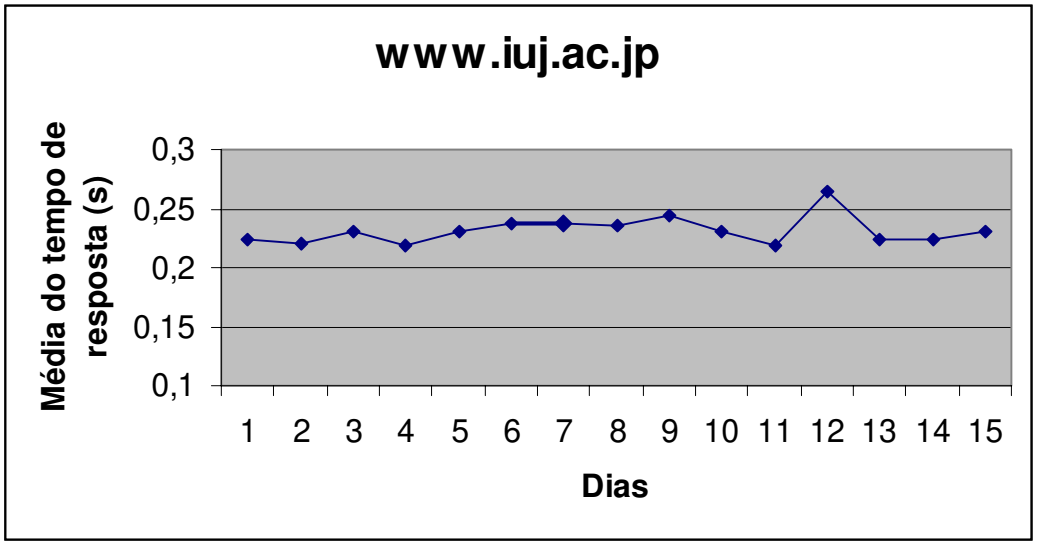

Figura 41 - Gráfico que demonstra a média do tempo de resposta para o envio de pacotes com 25 KB na rota que apresentou o pior resultado.

4.5 Sítio Laboratório Virtual

Para divulgação deste trabalho foi elaborado um sítio na Internet onde é possível acessar o experimento descrito obtendo informações adicionais sobre a pesquisa desenvolvida. O endereço é http://labvirtual.lme.usp.br/adriana/index.html (**Figura 42**).

Figura 42 - Interface do sítio do Laboratório Virtual implementado para divulgação da experiência elaborada como resultado experimental desta pesquisa.

4.5.1 Conclusões parciais

Nesta fase foram modificados os dois aplicativos já desenvolvidos com o objetivo de implementar um sistema de controle independente da plataforma. Na comunicação cliente/servido, o servidor, implementado em *LabVIEW®*, utiliza funções TCP/IP para estabelecer a comunicação com o cliente. O cliente, implementado em Java

pode ser executado em qualquer microcomputador com acesso a Internet e suporte à linguagem de programação Java. Desta forma temos um sistema cliente/servidor que oferece funcionalidade para os propósitos deste trabalho.

Através da monitoração das rotas descritas foi possível demonstrar a viabilidade do uso do laboratório virtual desenvolvido. A análise dos resultados obtidos, com uma velocidade de transmissão de dados satisfatória, mostra a funcionalidade e a potencialidade na utilização do Laboratório Virtual como aplicação nas áreas descritas no segundo capítulo deste trabalho.

Um sítio foi desenvolvido para divulgar as informações sobre o Laboratório Virtual.

5. CONCLUSÕES E PERSPECTIVAS FUTURAS

No primeiro capítulo foram apresentados aspectos introdutórios bem como objetivos e a justificativa para o desenvolvimento deste trabalho. A proposta global visa disponibilizar ensaios experimentais através de uma interface homem-máquina interativa para usuários conectados à Internet via *browser Web* em "tempo real". Destaca-se a possibilidade de conexão entre equipamentos e microcomputadores em rede local e remota via instrumentação virtual, permitindo o acesso interativo aos controles e o acompanhamento de experiências.

O segundo capítulo apresentou conceitos sobre sistemas que permitem o controle de instrumentação via Internet bem como uma descrição de alguns laboratórios virtuais pesquisados. Também foram apresentadas as tecnologias que tornam possível a implementação de ambientes para o controle e monitoração de experimentos de forma remota via rede de computadores. A estratégia de apresentação dessas tecnologias é baseada na instrumentação virtual que oferece a possibilidade de implementação de interfaces interativas e programação robusta bem como a Internet que permite conectar pessoas distribuídas por todo o mundo. Foi descrita a motivação no desenvolvimento de aplicações para popularizar o acesso a este meio de comunicação em diferentes áreas do conhecimento. Destaca-se o modelo de comunicação cliente/servidor utilizado pela maioria dos Laboratórios Virtuais pesquisados.

Foram apresentados, no capítulo 3, os materiais e métodos utilizados para o desenvolvimento deste trabalho onde se destaca a importância de cada elemento justificando as escolhas efetuadas.

No quarto capítulo foram apresentados os resultados experimentais obtidos bem como conclusões parciais sobre cada uma das etapas concluídas no desenvolvimento deste trabalho.

O sistema existente consiste na interligação de instrumentos de medida, placa de aquisição de dados e software que controla os instrumentos. Este controle pode ser realizado em rede local ou remota.

Foi implementado um VI (instrumento virtual desenvolvido em *LabVIEW®*) que controla o aparato experimental, baseado em um pacote de ferramentas da própria *National Instruments* (NI), que facilita a troca de comandos entre aplicações cliente/servidor como a que foi idealizada para este projeto. Este aplicativo comercial da NI, denominado *DataSocket®*, foi utilizado para estabelecer a comunicação entre o VI que controla o experimento (**VI-Servidor**) e um aplicativo desenvolvido em *Visual Basic* para enviar informações para um cliente em uma página *Web*. Este cliente também foi desenvolvido em *Visual Basic*; trata-se de um programa que utiliza a tecnologia *ActiveX*, o que permite seu funcionamento embutido em uma página *Web*. Temos um servidor, denominado **ServidorVB**, desenvolvido utilizando componentes *DataSocket®* para transmitir ao **VI-Servidor** os parâmetros necessários para executar a experiência e transmitir ao cliente solicitante as medidas adquiridas. O **ServidorVB** faz uso de funções TCP/IP para estabelecer uma interface de comunicação com o cliente e funções DSTP (*DataSocket Transfer Protocol*) que estabelece uma interface de comunicação com o **VI-Servidor**.

O *DataSocket®* versão 2.2, para esta experiência, não se mostrou eficiente. Sendo assim as funções do *DataSocket®* foram substituídas pelas funções TCP/IP no *LabVIEW®* dando origem a um novo programa servidor implementado em uma fase posterior do desenvolvimento do laboratório virtual.

Para suprir a necessidade de comunicação via protocolo TCP/IP e oferecer um software aplicativo flexível (independente de plataforma) implementou-se um novo programa cliente utilizando a linguagem de programação Java. Sendo assim, o sistema final possui um programa cliente desenvolvido com a linguagem de programação Java e um programa servidor, denominado **Servidor_GSIM**, implementado em *LabVIEW®* utilizando funções TCP/IP. A implementação feita em Java estabelece maior flexibilidade e versatilidade uma vez que o código pode ser portado e manipulado em várias plataformas. Esta linguagem também oferece maior grau de liberdade no sentido de manipular e definir diversas funções auxiliares para implementação de programas de simulação.

No endereço http://labvirtual.lme.usp.br/adriana/java/java.htm é possível acessar a experiência descrita neste trabalho bem como informações adicionais sobre o laboratório virtual.

Como perspectiva de atividades futuras dando continuidade aos resultados obtidos neste trabalho, pretende-se elaborar experimentos relacionados à eletrônica básica e disponibilizá-los aos alunos de nível médio de escolas técnicas colaborando, desta forma, com a educação a distância.

Existe ainda a possibilidade de desenvolvimento de ferramentas relacionadas à automação predial utilizando a instrumentação virtual e a Internet como ferramentas fundamentais para implementação de controle e monitoração de ambientes remotos.

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

1. Implementação do laboratório virtual para sistema operacional *Linux* fazendo uso dos novos padrões de protocolo, particularmente, o IP versão 6.
2. Estudar a viabilidade de implementação de um novo protocolo específico para a comunicação, via rede de longa distância, entre a instrumentação virtual e programas aplicativos de controle com o objetivo de aproveitar melhor os recursos existentes nas redes.
3. Desenvolver sistemas distribuídos de monitoração para redes de alta velocidade utilizando transmissão de imagens e som via Internet2.

.

REFERÊNCIAS BIBLIOGRÁFICAS

[ALLEN, 1998] ALLEN, ROBERT; "**The *Web*: interactive and multimedia education**", Computer Networks and ISDN systems, vol. 30, 1998.

[AL-MUHTADI, 2001] AL-MUHTADI, J.; MICKUNAS, D.; CAMPBELL, R., "**Wearable security services**", Distributed Computing Systems Workshop, 2001 International Conference, pp. 266 –271, 2001.

[ALTPETER, 1995] ALTPETER, F.; SALZMANN, CH.; GILLET, D.; LONGAHAMP, R.; "**A General Instrument for Real-time control and Data Acquisition**", 3rd IFAC/IFIP Workshop on Algorithms and Architecture for Real Time Control, Ostend, Belgium, June, 1995.

[ARPAIA, 2000] ARPAIA, P., BACCIGALUPI, A., CENNAMO, F., DUPONTE, P. "**A Measurement Laboratory on Geographic Network for Remote Test Experiments**", IEEE Transaction on Instrumentation and Measurement, vol. 49, no.5, October, 2000.

[BECK, 2001] BECK, THOMAS., "**Current trends in the design of automotive electronic systems Design**", Automation and Test Conference, 2001.

[BENETAZZO, 1999] BENETAZZO, L.; BERTOCCO, M.; FERRARIS, F.; FERRERO, A.; OFFELLI, C.; PARVIS, M.; PIURI, V. "**A *Web*-based distributed virtual educational laboratory**", Instrumentation and Measurement Technology Conference, 1999. Proceedings of the 16th IEEE , Volume: 3 , 1999.

[BERTOCCO, 1998] BERTOCCO, MATTEO; FERRARIS, FRANCO; OFFELLI, CARLO; PARVIS, MARCO; "A Client-Server Architecture for Distributed Measurement Systems", IEEE Transaction on Instrumentation and Measurement, vol. 47 no. 5, October, 1998.

[BILLE, 1997] BILLE, FULVIO; PUGLIESE, ROBERTO; "**Using WWW technology in a control system**", Nuclear Instruments and Methods in Physics Research A, 1997.

[BORGES, 1999] BORGES, ADRIANA P., FERNANDEZ, RODRIGO O. E RAMIREZ-FERNANDEZ, FRANCISCO J. **"Laboratório Virtual**". 7º SIICUSP - Simpósio Internacional de Iniciação Científica da Universidade de São Paulo, São Paulo, 1999.

[BORGES, 2000 a] BORGES, ADRIANA P., PERES-LISBOA, MAURICIO O., MORI, ALEXANDRE S. E RAMIREZ-FERNANDEZ, FRANCISCO J. "**A Graphical Interface to Link Virtual Instruments through a Web Browser**". Proceedings of SIBGRAPI 2000 - 13th Brazilian Symposium on Computer Graphics and Image Processing. IEEE Computer Society Press, 2000, pp.354.

[BORGES, 2000 b] BORGES, ADRIANA P., FERNANDEZ, RODRIGO O., PERES-LISBOA, MAURICIO O., MORI, ALEXANDRE S. E RAMIREZ-FERNANDEZ, FRANCISCO J. "**Virtual Laboratory Associated To Intelligent Instrumentation**". IES 2000 - Internet Education Science, Ucrânia, 2000.

[BROFFERIO, 1998] BROFFERIO, SERGIO CESARO, "**A University Distance Lesson System: Experiments, Services, and future developments**", IEEE Transaction On Education, vol. 41, nº 1, 1998, pp. 17-24.

[CHEN, 1999 a] CHEN, S.H.; RAMAKRISHNAN, V.; CHEN, R.; HU, S.Y.; ZHUANG, Y.; KO, C.C.; CHEN, BEM M.; "**A large-scale *Web*-based virtual Oscilloscope Laboratory experiment**", NUS – National University of Singapore, 1999.

[CHEN, 1999 b] CHEN, S.H.; RAMAKRISHNAN, V.; CHEN, R.; HU, S.Y.; ZHUANG, Y.; KO, C.C.; CHEN, BEM M.; "**Development of Remote Laboratory Experimentation through Internet**", Proceedings of the 1999 IEEE Hong Kong Symposium on Robotics and Control, Hong Kong, China, Volume II, pp.756-760, July 1999.

[COMER, 1988] COMER, DOUGLAS E., "**Interligação em rede com *TCP/IP***", Editora Campus – terceira edição, volume 1, São Paulo, 1988.

[CRISTALDI, 1999] CRISTALDI, LOREDANA; FERRERO, ALESSANDRO; PIURI, VINCENZO; "**Programmable Instruments, Virtual Instruments, and Distributed Measurement Systems: What is really useful, innovative and Technically Sound?**", IEEE Instrumentation & Measurement Magazine, September, 1999.

[FARRAR, 1997] FARRAR, BRIAN; "**Usando *ActiveX***", Editora Campus, 1997.

[FERNANDEZ, 2000] FERNANDEZ, RODRIGO O., BORGES, ADRIANA P., PEIXOTO, NATHALIA V. PERES-LISBOA, MAURICIO E RAMIREZ-FERNANDEZ, FRANCISCO J. "**Laboratório Virtual Aplicado À Educação A Distância**". SBIE2000 - XI Simpósio Brasileiro de Informática na Educação, 2000.

[FERRERO, 1999] FERRERO, ALESSANDRO; PIURI, VINCENSO; "**A Simulation Tool for Virtual Laboratory Experiments in a WWW Environment**", IEEE Transaction on Instrumentation, vol. 48, no. 3, June, 1999.

[FIELDLY, 2000] FIELDLY, T.A.; SHUR, M.S.; SHEN, H.; YTTERDAL, T. "**AIM-Lab: a system for remote characterization of electronic devices and circuits over the Internet**" , Devices, Circuits and Systems, 2000. pp. I43/1 -I43/6

[GOLDBERG, 2000] GOLDBERG, HAROLD, "**What is Virtual Instrumentation?**", IEEE Instrumentation & Measurement Magazine, pp. 10-13 December, 2000.

[HAHN, 1995] HAHN, HARLEY; STOUT, RICK "**Dominando a Internet**", Ed. Makron Books, São Paulo, 1995.

[HESSELINK, 1999] HESSELINK, LAMBERTUS; RIZAL, DHARMARUS; BJORSON, ERIC; "**Cyberlab, a new paradigm in Distance Learning**", Instrumentation Newsletter – National Instruments, third quarter 1999.

[HOON 1998] HOON, SUAT P., "**Conducting Experiments over the Internet**", Instrumentation Newsletter, National Instruments, vol. 10 no 4., 1998.

[JOHNSON, 1994] JOHNSON, GARY W., **"LabVIEW – graphical programming**", McGraw-Hill Inc,, 1994.

[LEE, 1999] LEE, KANG, B.; SCHNEEMAN, RICHARD D.; "**Internet-Based Distributed Measurement and Control Applications**", IEEE Transaction & Measurement Magazine, June, 1999.

[KATCHABAW, 1999] KATCHABAW, M.J., HOWARD S. L., LUTFIYYA, H. L., MARSHALL, A. D., "**Making distributed applications manageable through instrumentation**", The Journal of Systems and Software, no. 45, 1999, pp. 81-97.

[KO, 2000] KO C. C., CHEN B. M., CHEN S. H., RAMAKRISHNAN V., CHEN R., HU S. Y. ZHUANG Y., "**A large-scale Web-based virtual oscilloscope laboratory experiment",** Engineering Science and education Journal, april, 2000, pp. 69-76.

[MAGRABI, 1999] MAGRABI, FARAH; LOVELL, NIGEL H.; CELLER, BRANKO G.; "**A *Web*-based approach for electrocardiogram monitoring in the home**", International Journal of Medical Informatic, vol. 54, pp. 145-153, 1999.

[MALY, 1998] MALY, K.; OVERSTREET, C. M.; GONZÁLEZ, A .; DENBAR, M.; CUTARAN, R.; KARUNARATNE, N.; SRINIVAS, C.J.; "**Use *Web* technology**

for interactive remote instruction", Computer Network and ISDN Systems, vol. 30, 1998.

[NATIONAL INSTRUMENTS, 1998] "***LabVIEW* – user manual**", National Instruments, January, 1998.

[NATIONAL INSTRUMENTS, 1999] "***Aplications Note 127"***, National Instruments, June, 1999.

[MARINO, 2000] MARINO, P.; NOGUEIRA, J.; HERNANDEZ, H. "**Laboratory of virtual instrumentation for industrial electronics**", Proceedings of IEEE International Conference on Industrial Technology, Volume: 2 , 2000.

[MEYIR, 1994] G. C. MEYIR, A . W. VAN HERWAARDEN, "**Thermal Sensors**", Tu Delft Technisher Universitait, Delft, 1994.

[ORFALI, 1996] ORFALI, ROBERT; HALEY, DAN; EDWARDS, JERI; "**The essential client/server survival guide**", second edition, Wiley Computer Publishing Editora, 1996.

[OVERSTREET, 1999] OVERSTREET, J.W.; TZES, A. "**Internet-based client/server virtual instrument designs for real-time remote-access control engineering laboratory**", American Control Conference, Volume: 2 , 1999, pp. 1472 -1476

[PALOP, 2000] PALOP, JOSE M. GRIMA; TERUEL, JOSE M. ANDRES; "**Virtual Work Bench Electronic Instrumentation Teaching**", IEEE Transaction on Education, vol. 43, nº 1, February, 2000.

[PASSERINI, 2000] PASSERINI, KATIA; GRANGER, MARY J.; "**A developmental model for distance learning using the Internet**", Computer & Education, vol. 34, pp. 1-15, 2000.

[PATON 1998] PATON, BARRY E, "**Virtual Laser Lab - The Future of Distance Learning is Here!**", Instrumentation Newsletter, National Instruments, vol. 10 no 4., 1998.

[REGGIANI, 2000] REGGIANI, LÚCIA, "**A vida pelos bytes**", Revista InfoExame, ano 15, número 175, Outubro, 2000.

[SALZMANN, 1999] SALZMANN, CHRISTOPHE; "**Remote experimentation over the Internet**", Master Thesis presented at University of Florida, May, 1999.

[SALZMANN, 1998] SALZMANN, CHRISTOPHE; LATCHMAN, H. A .; GILLET, D.; CRISALLE, D.; "**Requirements for Real-time Laboratory Experimentation over the Internet**", International Conference on Engineering Education (ICEE98), August, 1998.

[SHEN, 1999] SHEN, HONG; XU, ZHENG; DALAGER, B.; KRISTIANSEN, V.; STROM, º; SHUR, MICHAEL S.; FJELDLY, TOR A .; LÜ, JIAN-QIANG; YTTERDAL, T.; "**Conducting Laboratory Experiments over the Internet**" , IEEE Transactions on education vol. 42 , no. 3, August 1999.

[SHEN, 2000] HONG SHEN; SHUR, M.S.; FJELDLY, T.A.; SMITH, K., "**Low-cost modules for remote engineering education: performing laboratory experiments over the Internet**", Frontiers in Education Conference, 2000, pp T1D/7.

[SHIN, 2000] SHIN, DONGIL; YOON, EN SUP; PARK, SANG JIN; LEE, EUY SOO; "***Web*-based interactive virtual laboratory system for unit operation and process systems engineering education**", Computer & Chemical Engineering, vol. 24, pp. 1381-1385, 2000.

[SPOELDER, 1999] SPOELDER, HANS J. W; "**Virtual Instrumentation and Virtual Environments**", IEEE Instrumentation & Measurement Magazine, September, 1999.

[STALLINGS, 1997] STALLINGS, WILLIAM; "**Local & Metropolitan Area Networks**", Prentice Hall Editora, 1993.

[STEGAWSKI, 1998] STEGAWSKI, MARCIN A.; SCHAUMANN, ROLF; "**A New Virtual-Instrumentation-Based Experimenting Environment for Undergraduate Laboratories with Application in Research and Manufacturing**", IEEE Transaction on Instrumentation and Measurement, vol. 47, nº 6, December 1998.

[TAN, 2001] TAN K. K., SOH C. Y., "**Instrumentation on the Internet**", Engineering Science and Education Journal, April 2001, pp. 61-67.

[TANENBAUM, 1987] TANENBAUM, ANDREW S., "**Redes de Computadores**", Editora Campus – terceira edição.

[TANER, 1997] TANER, A H; BRIGNELL, J. E.; "**Virtual instrumentation and intelligent sensors**", Sensors and Actuators A, vol. 61, pp. 427-430, 1997.

[VAUGHN, 1994] VAUGHN, LARRY T. "**Client/Server System Design & Implementation**", Ed. McGraw Hill, United State of America, 1994.

[WALLER, 2000] WALLER, JOHN C; FOSTER, NATHALIE; "**Training via the *Web*: a virtual instrument**", Computer & Education, vol. 35, pp. 161-197, March, 2000.

[WANG, 2000] WANG, CHANGTING; GAO, ROBERT X.; "**A Virtual Instrumentation System for Integrated Bearing Condition Monitoring**", IEEE Transaction on Instrumentation and Measurement, vol. 49, nº. 2, April, 2000.

Web Pages

[LINK 1] Disponível em <http://www.media.mit.edu/wearables/mithril/context/topic1.html.> Acesso em: 30 novembro 2000.
[LINK 2] AIM-Lab - Disponível em <http://www.nina.ecse.edu/shur/remote>. Acesso em novembro 1999.
[LINK 3] National University of Singapore – Disponível em: <http://vlab.ee.nus.edu.sg/vlab/ >. Acesso em: setembro 1999.
[LINK 4] Virtual Laser Lab – Disponível em: <http://vll.phys.dal.ca > . Acesso em: outubro 1999
[LINK 5] CyberLab – Disponível em: <http://cyberlab.stanford.edu>. Acesso em: setembro 1999.
[LINK 6] *National Institute of Standards and Technology* – Disponível em: <http://motion.aptd.nist.gov/P1451/ISADemo.htm>. Acesso em: junho 1999.
[LINK 7] Disponível em: <http://ni.com/labview >. Acesso em: setembro 1999.
[LINK 8] Disponível em < http://www.cern.ch >. Acesso em: novembro 2001.
[LINK 9] Disponível em: < http://www.estado.estadao.com.br/edicao/especial/internet/> . Acesso em: agosto 2001.
[LINK 10] Disponível em: < http://linuxgo.persogo.com.br/*socket*.html >. Acesso em: agosto 2001.
[LINK 11] Disponível em: < http://www.stanford.edu> . Acesso em: julho 1999.
[LINK 12] Disponível em: < http://msdn.microsoft.com/library/backgrnd/html/msdn_awhatis.htm >. Acesso em julho 2000.
[LINK 13] Disponível em: < http://*Activex*.com >. Acesso em julho 2000.
[LINK 14] Disponível em: < http://www.javaworld.com/javaworld/ >. Acesso em setembro 2000.
[LINK 15] Disponível em: <www.nessoft.com/pingplotter/>. Acesso em 20 dezembro 2001.

GLOSSÁRIO

ActiveX - Componentes utilizados para permitir a conexão via objetos ativos, com aplicações do lado servidor.

Applet - Um pequeno programa na linguagem Java que pode ser incluído em uma página em HTML.

Browser – Programa de aplicação cliente que permite acessar, por meio de uma interface gráfica (Windows, por exemplo), de maneira aleatória ou sistemática, informações diversas, contendo textos, imagens e gráficos, sons, etc. O acesso ao servidor remoto, que pode ou não estar ligado à Internet, pode ser feito via rede local ou modem.

CGI (Common Gateway Interface) - Aplicação servidora utilizada geralmente para processar solicitações do navegador (browser) através de formulários HTML, enviando o resultado em páginas dinâmicas HTML. Pode ser utilizado para conexão com outras aplicações e bancos de dados do servidor. Exemplos de linguagens de implementação: Perl, C e C++.

HTML (Hypertext Markup Language) - Linguagem padrão usada para escrever *páginas* de documentos para Web ou WWW. É uma variante da SGML (*Standard Generalized Markup Language*), bem mais fácil de aprender e usar, possibilitando preparar documentos com gráficos e links para outros documentos para visualização em sistemas que utilizam *Web*.

HTTP (HyperText Transfer Protocol) - Este protocolo é o conjunto de regras que permite a transferência de informações na Web e permite que os autores de páginas de hipertextos incluam comandos que possibilitem saltos para recursos e outros documentos disponíveis em sistemas remotos, de forma transparente para o usuário.

Java - Linguagem orientada a objetos, com sintaxe similar a C++, porém com biblioteca bastante distinta, que permite o desenvolvimento de aplicações e applets java. Gera código intermediário (byte codes) que são interpretados em tempo de execução, o que, juntamente com a sua biblioteca, torna a liguagem multi-plataforma, permitindo que seu código seja executado nas mais diversas máquinas e sistemas operacionais, sem a necessidade de adaptação. A Sun Microsystems, que inventou a linguagem Java, desenvolveu um browser para leitura dos *applets* e *classes*, e também um console para adaptação em outros navegadores. O Netscape Navigator bem como o Microsoft Internet Explorer já possibilitam a execução de applets Java. Mais informações no site Java da Sun Microsystems.

Mbps (Megabits por segundo) - Velocidade de tráfego de dados, equivalente a 10 milhões de bits por segundo.

Protocolo - Um conjunto de regras padronizado que especifica o formato, a sincronização, o seqüência e a verificação de erros em comunicação de dados. Uma descrição formal de formatos de mensagem e das regras que dois computadores devem obedecer ao trocar mensagens.

Sites – São as páginas na Internet.

Socket – Pequenos programas Conectores utilizados entre as aplicações e rede de computadores.

TCP/IP - Protocolo padrão da Internet.

Apêndice A

Código-fonte do programa cliente implementado em java

```
import java.awt.*;
import java.applet.*;
import java.applet.Applet;
import java.awt.event.*;
import java.io.*;
import java.net.*;
import java.text.*;
public class Cliente extends Applet implements ActionListener
{	boolean canvason, quadroon;
	Button start, reset;
	double[] vetor;
	double[][] x, y;
	int n;
	double ySize;
	Label quadro, quadro2;
	Plotador canvas;
	Plotador2 canvas2;
	String dados;
	TextField TensaoInicial, TensaoFinal, Incremento, Tempo, ptos, status;
	public void actionPerformed(ActionEvent evt)
	{	String command = evt.getActionCommand();
		if (command.equals("Enviar Dados"))
		{	double tensaoI = Double.valueOf(TensaoInicial.getText()).doubleValue();
			double tensaoF = Double.valueOf(TensaoFinal.getText()).doubleValue();
			double increm = Double.valueOf(Incremento.getText()).doubleValue();
			double tempo2= Double.valueOf(Tempo.getText()).doubleValue();
			if((tensaoI<0.5)||(tensaoI>10))
			{	status.setText("A tensão inicial deve variar de 0.5 V a 10 V");
			}
			else if((tensaoF<1)||(tensaoF>10))
			{	status.setText("A tensão final deve variar de 1 V a 10 V");
			}
			else if(tensaoF<(tensaoI+increm))
			{	status.setText("A tensão final deve ter um valor maior");
			}
			else if((increm<0.5)||(increm>1))
			{	status.setText("A tensão de incremento deve variar de 0.5 V a 1 V");
			}
			else if((tempo2<0.5)||(tempo2>3))
			{	status.setText("O tempo de amostra deve variar de 0.5 s a 3 s");
			}
			else
			{		dados = Conecta(PreparaDados());
				if (dados.charAt(0) == '@')
				{	dados = dados.substring(1);
					status.setText(dados);
				}
				else
```

```
                                        {       Formata();
                                                PlotaGrafico();
                                        }
                                }
                }
                else if (command.equals("Reset"))
                        Reseta();
        }
        public String Conecta(String dataout)
        {       status.setText("Conectando...");
                String serverurl = "143.107.160.136";
                int serverport = 17777;
                Socket socket = null;
                String tchau = null;
                try
                {       status.setText("Conectando");
                        socket = new Socket(serverurl, serverport);
                        socket.setSoTimeout(12000000);
                        status.setText("Enviando dados...");
                        InputStreamReader inputstreamreader = new
InputStreamReader(socket.getInputStream());
                        BufferedReader in = new BufferedReader(inputstreamreader);
                        PrintWriter out = new PrintWriter(socket.getOutputStream(),true);
                        int outlength = dataout.length();
                        DecimalFormat myFormatter = new DecimalFormat("0000");
                        out.println(myFormatter.format(outlength));
                        out.println(dataout);
                        status.setText("Recebendo dados...");
                        StringBuffer datain = new StringBuffer();
                        datain.append(in.readLine());
                        status.setText("Encerrando a conexão...");
                        tchau = datain.toString();
                        in.close();
                        inputstreamreader.close();
                        out.close();
                        socket.close();
                }
                catch(UnknownHostException unhe)
                {       tchau = ("@Erro desconhecido.");
                }
                catch(InterruptedIOException intioe)
                {       tchau = ("@Servidor ocupado. Tente mais tarde.");
                }
                catch(IOException ioe)
                {       tchau = ("@Servidor inacessível. Tente mais tarde.");
                }
                return tchau;
        }
        public void Formata()
        {       status.setText("Formatando dados...");
                int tamanho = dados.length();
                n = Integer.valueOf(ptos.getText()).intValue();
                double[] vetor = new double[(n + 1) * 10];
                int k = 0;
```

```
            int inicio, fim;
            for (int i = 0; i < tamanho; i++)
            {
                  if (i == 0)
                  {     inicio = 0;
                        for (int j = (inicio + 1); j < tamanho; j++)
                        {     if (dados.charAt(j) == '\t')
                              {     fim = j;
                                    String trecho = dados.substring(inicio, fim);
                                    Double valor = Double.valueOf(trecho);
                                    vetor[k] = valor.doubleValue();
                                    k++;
                                    j = tamanho;
                              }
                        }
                  }
                  else
                  {     if (dados.charAt(i) == '\t')
                        {     inicio = i;
                              for (int j = (inicio + 1); j < tamanho; j++)
                              {           if (dados.charAt(j) == '\t')
                                    {     fim = j;
                                          String trecho = dados.substring(inicio, fim);
                                          Double valor = Double.valueOf(trecho);
                                          vetor[k] = valor.doubleValue();
                                          k++;
                                          j = tamanho;
                                    }
                              }
                        }
                  }
            }
            x = new double[n][5];
            y = new double[n][5];
            int m = 0;
            for(int i = 0; i < n; i++)
            {     for(int j = 0; j < 5; j++)
                  {     x[i][j] = vetor[m];
                        m++;
                        y[i][j] = vetor[m];
                        m++;
                  }
            }
}

public void init()
{    MontaInterface();
}

public void LimpaTela()
{     if (canvason == true)
      {     canvason = false;
      }
      if (quadroon == true)
```

```
            {           remove(quadro);
            remove(quadro2);
            quadroon = false;
            }
}
public void MontaInterface()
{           Font FontePadrao = new Font("Arial", Font.BOLD, 12);
            Font FonteMaior = new Font("Arial", Font.BOLD, 20);
            Color CorPadrao = Color.white;
            Label labum, labdois, labtres, labquatro;
            Label labptos, labstatus, labseis;
            //resize(430,450);
            resize(770,480);
            setBackground(new Color(255,10,40));
            setLayout(null);
            labum = new Label("Tensão Inicial(V):", 1);
            labum.setForeground(CorPadrao);
            labum.setFont(FontePadrao);
            TensaoInicial = new TextField("0.00");
            TensaoInicial.setBackground(CorPadrao);
            TensaoInicial.setFont(FontePadrao);
            labdois = new Label("Tensão Final(V):", 1);
            labdois.setForeground(CorPadrao);
            labdois.setFont(FontePadrao);
            TensaoFinal = new TextField("0.00");
            TensaoFinal.setBackground(CorPadrao);
            TensaoFinal.setFont(FontePadrao);
            labtres = new Label("Incremento(V):", 1);
            labtres.setForeground(CorPadrao);
            labtres.setFont(FontePadrao);
            Incremento = new TextField("0.00");
            Incremento.setBackground(CorPadrao);
            Incremento.setFont(FontePadrao);
            labquatro = new Label("Tempo de Amostra(s):", 1);
            labquatro.setForeground(CorPadrao);
            labquatro.setFont(FontePadrao);
            Tempo = new TextField("0.00");
            Tempo.setBackground(CorPadrao);
            Tempo.setFont(FontePadrao);
            labptos = new Label("Ptos/Curva:", 1);
            labptos.setForeground(CorPadrao);
            labptos.setFont(FontePadrao);
            ptos = new TextField("25");
            ptos.setBackground(CorPadrao);
            ptos.setFont(FontePadrao);
            start = new Button("Enviar Dados");
            start.setFont(FontePadrao);
            start.addActionListener(this);
            reset = new Button("Reset");
            reset.setFont(FontePadrao);
            reset.addActionListener(this);
            quadro = new Label("",1);
            quadro.setBackground(CorPadrao);
            quadro.setFont(FonteMaior);
```

```
                quadro2 = new Label("",1);
                quadro2.setBackground(CorPadrao);
                quadro2.setFont(FonteMaior);
                labstatus = new Label("Status:", 1);
                labstatus.setForeground(CorPadrao);
                labstatus.setFont(FontePadrao);
                status = new TextField("Não conectado.");
                status.setBackground(CorPadrao);
                status.setFont(FontePadrao);
                add(labum); add(TensaoInicial); add(labdois); add(TensaoFinal);
                add(labtres); add(Incremento); add(labquatro); add(Tempo);
                add(labptos); add(ptos); add(start); add(reset);
                add(quadro); add(labstatus); add(status);add(quadro2);
                canvason = false; quadroon = true;
                labstatus.setBounds(200, 10, 48, 20);
                status.setBounds(250, 10, 330, 20);
                labum.setBounds(200, 40, 100, 20);
                TensaoInicial.setBounds(300, 40, 40, 20);
                labdois.setBounds(196, 70, 100, 20);
                TensaoFinal.setBounds (300, 70, 40, 20);
                labtres.setBounds(440, 40, 95, 20);
                Incremento.setBounds(540, 40, 40, 20);
                labquatro.setBounds(400, 70, 130, 20);
                Tempo.setBounds(540, 70, 40, 20);
                start.setBounds(120, 420, 130, 20);
                reset.setBounds(500, 420, 130, 20);
                //quadro.setBounds(10, 100, 410, 310);
                quadro.setBounds(10, 100, 370, 300);
                //8888888888
                quadro2.setBounds(390, 100, 370, 300);
        }
        public String PreparaDados()
        {       StringBuffer ent = new StringBuffer();
                ent.append(TensaoInicial.getText() + "\t");
                ent.append(TensaoFinal.getText() + "\t");
                ent.append(Incremento.getText() + "\t");
                ent.append(Tempo.getText() + "\t");
           return ent.toString();
        }
        public void PlotaGrafico()
        {       String[] Labels = new String[4];
                Labels[0] = "Tensão X Tempo";
                Labels[1] = "";
                Labels[2] = "Tempo(s)";
                Labels[3] = "Tensão(V)";
                LimpaTela();
                double tensaoI = Double.valueOf(TensaoInicial.getText()).doubleValue();
                double tensaoF = Double.valueOf(TensaoFinal.getText()).doubleValue();
                double increm = Double.valueOf(Incremento.getText()).doubleValue();
                double tempo2= Double.valueOf(Tempo.getText()).doubleValue();
                double loop = (tensaoF-tensaoI)/increm;
                double escala=loop*tempo2;
                canvas = new Plotador(loop,x[0][0],y[0][0],x[0][1],y[0][1],x, y, Labels, 4,
escala,tensaoF,tempo2);
```

```
                add(canvas);
                canvas.setBounds(10, 100, 370, 300);
                Labels[0] = "Temperatura X Tempo";
                Labels[1] = "";
                Labels[2] = "Tempo(s)";
                Labels[3] = "Temperatura(ºC)";
                LimpaTela();
                canvas2 = new Plotador2(loop,x[0][0],y[0][0],x[0][1],y[0][1],x, y, Labels, 4,
escala,tensaoF,tempo2);
                add(canvas2);
                canvas2.setBounds(390, 100, 370, 300);
                canvas = true;
                status.setText("Não conectado.");
        }
        public void Reseta()
        {       LimpaTela();
                remove(canvas);
                remove(canvas2);
                remove(quadro2);
                remove(quadro);
                add(quadro);
                add(quadro2);
                quadroon = true;
                TensaoInicial.setText("0.00");
                TensaoFinal.setText("0.00");
                Incremento.setText("0.00");
                Tempo.setText("0.00");
                ptos.setText("25");
                status.setText("Não conectado.");
        }

}
```

//Plotador.java

```
import java.awt.*;
class Plotador extends java.awt.Canvas
{     double[][]x;
      String xLabel,yLabel,title, subtitle;
      double[][]y;
      int numberOfPlots;
      int plotOption;
      double xDisplayMin;
      double xDisplayMax;
      double yDisplayMin;
      double yDisplayMax;
      double xDisplayRange;
      double yDisplayRange;
      int xOrderOfMag;
      int yOrderOfMag;
      int limitOption=0;
      int xCase;
      int yCase;
      double xTick;
```

```
	double yTick;
	int[][] xInt;
	int[][] yInt;
	int scalingOption;
	int xSize;
	int ySize;
	double eValue = 2.3025851;
	double escala;
	double tensaoF;
	double tempo2;
	double xMax;
	double temp;
	double t1;
	double t2;
	double t3;
	double[] teste;
	double loop;
	double tempantes;
	double t1antes;
	double t2antes;
	double t3antes;
	double tempdepois;
	double t1depois;
	double t2depois;
	double t3depois;
	double tempoantes;
	double tempodepois;
	Plotador(double loop,double temp,double t1,double t2,double
t3,double[][] x, double[][] y, String[] labels, int numberOfPlots,
double escala,double tensaoF,double tempo2)
	{	this.numberOfPlots = numberOfPlots;
		this.y = y;
		this.x = x;
		this.xLabel = labels[2];
		this.yLabel = labels[3];
		this.title = labels[0];
		this.subtitle = labels[1];
		this.escala = escala;
		this.tensaoF=tensaoF;
		this.tempo2=tempo2;
		this.temp=temp;
		this.t1=t1;
		this.t2=t2;
		this.t3=t3;
		this.loop=loop;
	}
	public void autoXScaling()
	{	this.xSize = getSize().width;
		double xMax=escala;
		double xMin=1;
		int[][] xi = new int[x.length][numberOfPlots];
		xCase = 1;
		xOrderOfMag=(int)(Math.log(xMax)/eValue);
		if (Math.log(xMax-xMin)<0) xOrderOfMag = xOrderOfMag-1;
```

```
            xTick=tempo2;
            xDisplayMin=0;
            xDisplayMax = 0.0;
            while (xDisplayMax<(1.01*xMax))
            {     xDisplayMax=xDisplayMax +
(xTick*Math.exp(xOrderOfMag*eValue));
            }
            xDisplayRange=xDisplayMax - xDisplayMin;
            for (int i=0; i<x.length;i++)
            {     for (int j=0; j<numberOfPlots; j++)
                  {     xi[i][j] =
(int)(((x[i][j]/xDisplayRange)*0.8*xSize)+(0.075*xSize));
                  }
            }
            this.xInt = xi;
      }
      public void autoYScaling()
      {     this.ySize = getSize().height;
            double yMax=tensaoF;
            double yMin=1;
            int[][] yi = new int[x.length][numberOfPlots];
            yCase = 1;
                  yOrderOfMag=(int)(Math.log(yMax)/eValue);
                  if (Math.log(yMax)<0) yOrderOfMag = yOrderOfMag-1;
                  if (((Math.log(yMax)/eValue)-yOrderOfMag)<0.5)
                  yTick =0.5;
                  else yTick = 1;
                  yDisplayMin=0;
                  yDisplayMax = 0.0;
                  while (yDisplayMax<(1.01*yMax))
                  {     yDisplayMax=yDisplayMax +
(yTick*Math.exp(yOrderOfMag*eValue));
                  }
                  yDisplayRange=yDisplayMax - yDisplayMin;
                  for (int i=0;i<x.length;i++)
                  {     for (int j=0;j<numberOfPlots;j++)
                        {     yi[i][j]=(int)(ySize -
(((y[i][j]/yDisplayRange)*0.8*ySize)+(0.075*ySize)));
                        }
                  }
            this.yInt = yi;
      }
      public void paint(Graphics g)
      {
            autoXScaling();
            autoYScaling();
            g.setColor(Color.white);
            g.fillRect(0,0,(int)xSize,(int)ySize);
                  double Tick1;
                  double Tick2;
                  int yAxisCoord = (int)(0.075*xSize);
                  Tick1=xTick;
                  tempoantes=xTick;
```

```
                int xCoord =
(int)((((Tick1*Math.exp(xOrderOfMag*eValue))/xDisplayRange)*0.8*xSize)+
(0.075*xSize));
            Tick2=temp/10;
            tempantes=Tick2;
            int yCoord = (int)(ySize -
((((Tick2*Math.exp(yOrderOfMag*eValue))/yDisplayRange)*0.8*ySize)+(0.07
5*ySize)));             //desenhar tensão1
                g.setColor(Color.blue);
                Tick2=t1;
                t1antes=t1;
                yCoord = (int)(ySize -
((((Tick2*Math.exp(yOrderOfMag*eValue))/yDisplayRange)*0.8*ySize)+(0.07
5*ySize)));

      //g.drawLine((int)(0.075*xSize),(int)(0.925*ySize),xCoord,yCoord)
;
            //desenhar tensao2
                g.setColor(Color.orange);
                Tick2=t2/10;
                t2antes=t2/10;
                yCoord = (int)(ySize -
((((Tick2*Math.exp(yOrderOfMag*eValue))/yDisplayRange)*0.8*ySize)+(0.07
5*ySize)));
      //g.drawLine((int)(0.075*xSize),(int)(0.925*ySize),xCoord,yCoord)
;
                //desenhar t3
                g.setColor(Color.red);
                Tick2=t3;
                t3antes=t3;
                yCoord = (int)(ySize -
((((Tick2*Math.exp(yOrderOfMag*eValue))/yDisplayRange)*0.8*ySize)+(0.07
5*ySize)));
      //g.drawLine((int)(0.075*xSize),(int)(0.925*ySize),xCoord,yCoord)
;
                int p = 0;
                int v=0;
                int f=0;
                int s=2;
                while(v < loop)
                {     v++;
                      tempdepois=x[f][s];
                      t1depois=y[f][s];
                      s++;
                      if(s>4)
                      {
                            f++;
                      }
                      if(s>4)
                      {
                            s=0;
                      }
                      t2depois=x[f][s]/10;
                      t3depois=y[f][s];
```

```
                        s++;
                        if(s>4)
                        {
                              f++;
                        }
                        if(s>4)
                        {
                              s=0;
                        }
                        tempodepois=tempoantes+tempo2;
                        int xCoord1 =
(int)((((tempoantes*Math.exp(xOrderOfMag*eValue))/xDisplayRange)*0.8*xS
ize)+(0.075*xSize));
                        int xCoord2 =
(int)((((tempodepois*Math.exp(xOrderOfMag*eValue))/xDisplayRange)*0.8*x
Size)+(0.075*xSize));

                        int yCoord1 = (int)(ySize -
((((tempantes*Math.exp(yOrderOfMag*eValue))/yDisplayRange)*0.8*ySize)+(
0.075*ySize)));
                        int yCoord2 = (int)(ySize -
((((tempdepois*Math.exp(yOrderOfMag*eValue))/yDisplayRange)*0.8*ySize)+
(0.075*ySize)));
                        //desenhar tensão 1
                        g.setColor(Color.blue);
                        yCoord1 = (int)(ySize -
((((t1antes*Math.exp(yOrderOfMag*eValue))/yDisplayRange)*0.8*ySize)+(0.
075*ySize)));
                        yCoord2 = (int)(ySize -
((((t1depois*Math.exp(yOrderOfMag*eValue))/yDisplayRange)*0.8*ySize)+(0
.075*ySize)));
                        g.drawLine(xCoord1,yCoord1,xCoord2,yCoord2);
                        //desenhar tensão 2
                        g.setColor(Color.orange);
                        yCoord1 = (int)(ySize -
((((t2antes*Math.exp(yOrderOfMag*eValue))/yDisplayRange)*0.8*ySize)+(0.
075*ySize)));
                        yCoord2 = (int)(ySize -
((((t2depois*Math.exp(yOrderOfMag*eValue))/yDisplayRange)*0.8*ySize)+(0
.075*ySize)));
                        g.drawLine(xCoord1,yCoord1,xCoord2,yCoord2);
                        //desenhar tensão 3
                        g.setColor(Color.red);
                        yCoord1 = (int)(ySize -
((((t3antes*Math.exp(yOrderOfMag*eValue))/yDisplayRange)*0.8*ySize)+(0.
075*ySize)));
                        yCoord2 = (int)(ySize -
((((t3depois*Math.exp(yOrderOfMag*eValue))/yDisplayRange)*0.8*ySize)+(0
.075*ySize)));
                        g.drawLine(xCoord1,yCoord1,xCoord2,yCoord1);
                        g.drawLine(xCoord2,yCoord1,xCoord2,yCoord2);
                        tempoantes=tempodepois;
                        tempantes=tempdepois;
                        t1antes=t1depois;
```

```
                        t2antes=t2depois;
                        t3antes=t3depois;
                  }
      g.setColor(Color.black);
      g.drawLine((int)(0.075*xSize),(int)(0.125*ySize),(int)(0.075*xSiz
e),(int)(0.925*ySize));
            Font titleFont = new Font("Arial",Font.BOLD,15);
            g.setFont(titleFont);
            g.drawString(title, (int)(0.35*xSize),(int)(0.10*ySize));
            Font subtitleFont = new Font("Arial",Font.BOLD,12);
            g.setFont(subtitleFont);
            g.drawString(subtitle,
(int)(0.35*xSize),(int)(0.085*ySize));
            Font labelFont = new Font("Arial",Font.BOLD,10);
            g.setFont(labelFont);
            g.drawString(yLabel, (int)(0.005*xSize),(int)(0.06*ySize));
            g.drawString(xLabel, (int)(0.45*xSize),(int)(0.985*ySize));
             yAxisCoord=0;
            int xAxisCoord=0;
            if (yCase == 1)
            {     xAxisCoord=(int)(0.925*ySize);

      g.drawLine((int)(0.075*xSize),xAxisCoord,(int)(0.875*xSize),xAxis
Coord);
            }
            if (yOrderOfMag == 1)
            {     yTick = 10*yTick;
                  yOrderOfMag = 0;
            }
      yAxisCoord =  (int)(0.075*xSize);
            int TickCoord;
            int OOMCoord;
            if (yCase<3)
                  {     TickCoord = (xAxisCoord+(int)(0.03*ySize));
                        OOMCoord = (xAxisCoord+(int)(0.045*ySize));
                  }
                  else
                  {     TickCoord =(xAxisCoord-(int)(0.015*ySize));
                        OOMCoord = (xAxisCoord-(int)(0.03*ySize));
                  }
            if(tempo2>10)
            {
                  xTick=xTick/10;
                  xOrderOfMag = 0;
            }
            double teste;
            double Tick;
            switch (xCase)
            {     case 1:
                        {Tick = 0;
                        while
((Tick*Math.exp(xOrderOfMag*eValue))<(1.01*xDisplayMax))
                        {
```

```
                              xCoord =
(int)((((Tick*Math.exp(xOrderOfMag*eValue))/xDisplayRange)*0.8*xSize)+(
0.075*xSize));
                              g.drawLine(xCoord,(int)(xAxisCoord-
(0.005*ySize)),xCoord,(int)(xAxisCoord+(0.005*ySize)));
                              teste=(int)(Tick*100);
                              teste=teste/100;
                              Tick=(double)teste;

      g.drawString(Double.toString(Tick),(xCoord-
(int)(0.015*ySize)),TickCoord);
                              Tick = Tick + tempo2;
                        }
                        }
                  }
      TickCoord = (int)(yAxisCoord-(0.07*ySize));
      switch (yCase)
            {     case 1:
                        {Tick = 0.0;
                        while ((Tick*Math.exp(yOrderOfMag*eValue)) <
(1.01*yDisplayMax))
            {     yCoord = (int)(ySize -
((((Tick*Math.exp(yOrderOfMag*eValue))/yDisplayRange)*0.8*ySize)+(0.075
*ySize)));
                              g.drawLine((int)(yAxisCoord-
(0.005*xSize)),yCoord,(int)(yAxisCoord+(0.005*xSize)),yCoord);
      g.drawString(Double.toString(Tick),TickCoord,yCoord+(int)(0.015*y
Size));
                              Tick = Tick + yTick;
                        }
                        }
            case 2:
                        if (yDisplayMin<0)
                        {     Tick = -yTick;
                              while
((Tick*Math.exp(yOrderOfMag*eValue)) > (1.01*yDisplayMin))
                              {     yCoord = (int)(ySize -
(((((Tick*Math.exp(yOrderOfMag*eValue))-
yDisplayMin)/(yDisplayRange))*0.8*ySize)+(0.075*ySize)));
                                    g.drawLine((int)(yAxisCoord-
(0.005*xSize)),yCoord,(int)(yAxisCoord+(0.005*xSize)),yCoord);
      g.drawString(Double.toString(Tick),TickCoord,yCoord+(int)(0.015*y
Size));
                                    Tick = Tick - yTick;
                              }
                              Tick = 0.0;
                              while
((Tick*Math.exp(yOrderOfMag*eValue)) < (1.01*yDisplayMax))
                              {     yCoord = (int)(ySize -
(((((Tick*Math.exp(yOrderOfMag*eValue))-
yDisplayMin)/(yDisplayRange))*0.8*ySize)+(0.075*ySize)));
                                    g.drawLine((int)(yAxisCoord-
(0.005*xSize)),yCoord,(int)(yAxisCoord+(0.005*xSize)),yCoord);
```

```
      g.drawString(Double.toString(Tick),TickCoord,yCoord+(int)(0.015*y
Size));
                                    Tick = Tick + yTick;
                              }
                        }
                        else
                        {     Tick = 0.0;
                              while
((Tick*Math.exp(yOrderOfMag*eValue)) < (0.99*yDisplayMin))
                              {     Tick = Tick + yTick;
                              }
                              while
((Tick*Math.exp(yOrderOfMag*eValue)) < (1.01*yDisplayMax))
                              {     yCoord = (int)(ySize -
(((((Tick*Math.exp(yOrderOfMag*eValue))-
yDisplayMin)/(yDisplayRange))*0.8*ySize)+(0.075*ySize)));
                                    g.drawLine((int)(yAxisCoord-
(0.005*xSize)),yCoord,(int)(yAxisCoord+(0.005*xSize)),yCoord);

      g.drawString(Double.toString(Tick),TickCoord,yCoord+(int)(0.015*y
Size));
                                    Tick = Tick + yTick;
                              }
                        }
                  case 3:
                        {Tick = 0.0;
                        while ((Tick*Math.exp(yOrderOfMag*eValue)) >
(0.99*yDisplayMax))
                              {     Tick = Tick - yTick;
                              }
                        while ((Tick*Math.exp(yOrderOfMag*eValue)) >
(1.01*yDisplayMin))
                        {     yCoord = (int)(ySize -
(((((Tick*Math.exp(yOrderOfMag*eValue))-
yDisplayMin)/yDisplayRange)*0.8*ySize)+(0.075*ySize)));
                              g.drawLine((int)(yAxisCoord-
(0.005*xSize)),yCoord,(int)(yAxisCoord+(0.005*xSize)),yCoord);

      g.drawString(Double.toString(Tick),TickCoord,yCoord+(int)(0.015*y
Size));
                              Tick = Tick - yTick;
                        }
                        }
                  case 4:
                        {Tick = 0.0;
                        while ((Tick*Math.exp(yOrderOfMag*eValue)) >
(1.01*yDisplayMin))
                        {     yCoord = (int)(ySize -
(((1+((Tick*Math.exp(yOrderOfMag*eValue))/yDisplayRange))*0.8*ySize)+(0
.075*ySize)));
                              g.drawLine((int)(yAxisCoord-
(0.005*xSize)),yCoord,(int)(yAxisCoord+(0.005*xSize)),yCoord);
      g.drawString(Double.toString(Tick),TickCoord,yCoord+(int)(0.015*y
Size));
```

```
                                Tick = Tick - yTick;
                            }
                        }
                        break;
                    }
            }
}
```

//Plotador2.java

```
import java.awt.*;
class Plotador2 extends java.awt.Canvas
{     double[][]x;
      String xLabel,yLabel,title, subtitle;
      double[][]y;
      int numberOfPlots;
      int plotOption;
      double xDisplayMin;
      double xDisplayMax;
      double yDisplayMin;
      double yDisplayMax;
      double xDisplayRange;
      double yDisplayRange;
      int xOrderOfMag;
      int yOrderOfMag;
      int limitOption=0;
      int xCase;
      int yCase;
      double xTick;
      double yTick;
      int[][] xInt;
      int[][] yInt;
      int scalingOption;
      int xSize;
      int ySize;
      double eValue = 2.3025851;
      double escala;
      double tensaoF;
      double tempo2;
      double xMax;
      double temp;
      double t1;
      double t2;
      double t3;
      double[] teste;
      double loop;
      double tempantes;
      double t1antes;
      double t2antes;
      double t3antes;
      double tempdepois;
      double t1depois;
      double t2depois;
      double t3depois;
      double tempoantes;
```

```
      double tempodepois;
Plotador2(double loop,double temp,double t1,double t2,double
t3,double[][] x, double[][] y, String[] labels, int numberOfPlots,
double escala,double tensaoF,double tempo2)
      {     this.numberOfPlots = numberOfPlots;
            this.y = y;
            this.x = x;
            this.xLabel = labels[2];
            this.yLabel = labels[3];
            this.title = labels[0];
            this.subtitle = labels[1];
            this.escala = escala;
            this.tensaoF=tensaoF;
            this.tempo2=tempo2;
            this.temp=temp;
            this.t1=t1;
            this.t2=t2;
            this.t3=t3;
            this.loop=loop;
      }

      public void autoXScaling()
      {     this.xSize = getSize().width;
            double xMax=escala;
            double xMin=1;
            int[][] xi = new int[x.length][numberOfPlots];
            xCase = 1;
            xOrderOfMag=(int)(Math.log(xMax)/eValue);
            if (Math.log(xMax-xMin)<0) xOrderOfMag = xOrderOfMag-1;
            if(tensaoF==11)
            {
                  xTick=tempo2/10;
            }
            else xTick=tempo2;


            if(escala>10)
            {     xTick=tempo2/10;
            }
            xDisplayMin=0;
            xDisplayMax = 0.0;
            while (xDisplayMax<(1.01*xMax))
            {
                  xDisplayMax=xDisplayMax +
(xTick*Math.exp(xOrderOfMag*eValue));
            }
            xDisplayRange=xDisplayMax - xDisplayMin;
            for (int i=0; i<x.length;i++)
            {     for (int j=0; j<numberOfPlots; j++)
                  {     xi[i][j] =
(int)(((x[i][j]/xDisplayRange)*0.8*xSize)+(0.075*xSize));
                  }
            }
```

```
            this.xInt = xi;
      }
      public void autoYScaling()
      {
            this.ySize = getSize().height;
            double yMax=20;
            double yMin=1;
            for (int i=0;i<y.length;i++)
            {     for (int j=0;j<numberOfPlots;j++)
                  {     if (y[i][j]>yMax) yMax=y[i][j];
                        if (y[i][j]<yMin) yMin=y[i][j];
                  }
            }
            int[][] yi = new int[x.length][numberOfPlots];
            yCase = 1;
                  yOrderOfMag=(int)(Math.log(yMax)/eValue);
                  if (Math.log(yMax)<0) yOrderOfMag = yOrderOfMag-1;
                  if (((Math.log(yMax)/eValue)-yOrderOfMag)<0.5)
                  yTick =0.5;
                  else yTick = 1;
                  yDisplayMin=0;
                  yDisplayMax = 0.0;
                  while (yDisplayMax<(1.01*yMax))
                  {     yDisplayMax=yDisplayMax +
(yTick*Math.exp(yOrderOfMag*eValue));
                  }
                  yDisplayRange=yDisplayMax - yDisplayMin;

                  for (int i=0;i<x.length;i++)
                  {     for (int j=0;j<numberOfPlots;j++)
                        {     yi[i][j]=(int)(ySize -
(((y[i][j]/yDisplayRange)*0.8*ySize)+(0.075*ySize)));
                        }
                  }
            this.yInt = yi;

      }
      public void paint(Graphics g)
      {
            autoXScaling();
            autoYScaling();
            g.setColor(Color.white);
            g.fillRect(0,0,(int)xSize,(int)ySize);
                  //Desenhar linha de temperatura
                  g.setColor(Color.magenta);
                  double Tick1;
                  double Tick2;
                  int yAxisCoord = (int)(0.075*xSize);
                  Tick1=xTick;
                  tempoantes=xTick;
                  int xCoord =
(int)((((Tick1*Math.exp(xOrderOfMag*eValue))/xDisplayRange)*0.8*xSize)+
(0.075*xSize));
                  Tick2=temp/10;
```

```
                  tempantes=Tick2;
                  int yCoord = (int)(ySize -
((((Tick2*Math.exp(yOrderOfMag*eValue))/yDisplayRange)*0.8*ySize)+(0.07
5*ySize)));
      //g.drawLine((int)(0.075*xSize),(int)(0.925*ySize),xCoord,yCoord)
;
                  int p = 0;
                  int v=0;
                  int f=0;
                  int s=2;
                  int rv=1;
                  while(v < loop)
                  {
                        v++;
                        tempdepois=x[f][s]/10;
                        t1depois=y[f][s];
                        s++;
                        if(s>4)
                        {
                              f++;
                        }
                        if(s>4)
                        {
                              s=0;
                        }
                        t2depois=x[f][s];
                        t3depois=y[f][s];
                        s++;
                        if(s>4)
                        {
                              f++;
                        }
                        if(s>4)
                        {
                              s=0;
                        }

                        tempodepois=tempoantes+tempo2;
                        int xCoord1 =
(int)((((tempoantes*Math.exp(xOrderOfMag*eValue))/xDisplayRange)*0.8*xS
ize)+(0.075*xSize));
                        int xCoord2 =
(int)((((tempodepois*Math.exp(xOrderOfMag*eValue))/xDisplayRange)*0.8*x
Size)+(0.075*xSize));

                        //temperatura
                        g.setColor(Color.magenta);
                        int yCoord1 = (int)(ySize -
((((tempantes*Math.exp(yOrderOfMag*eValue))/yDisplayRange)*0.8*ySize)+(
0.075*ySize)));
                        int yCoord2 = (int)(ySize -
((((tempdepois*Math.exp(yOrderOfMag*eValue))/yDisplayRange)*0.8*ySize)+
(0.075*ySize)));
                        g.drawLine(xCoord1,yCoord1,xCoord2,yCoord2);
```

```
                        tempoantes=tempodepois;
                        tempantes=tempdepois;
                        t1antes=t1depois;
                        t2antes=t2depois;
                        t3antes=t3depois;
                  }
                  g.setColor(Color.blue);
                  g.setColor(Color.black);
      g.drawLine((int)(0.075*xSize),(int)(0.125*ySize),(int)(0.075*xSiz
e),(int)(0.925*ySize));
            Font titleFont = new Font("Arial",Font.BOLD,15);
            g.setFont(titleFont);
            g.drawString(title, (int)(0.35*xSize),(int)(0.10*ySize));
            Font subtitleFont = new Font("Arial",Font.BOLD,12);
            g.setFont(subtitleFont);
            g.drawString(subtitle,
(int)(0.35*xSize),(int)(0.085*ySize));
            Font labelFont = new Font("Arial",Font.BOLD,10);
            g.setFont(labelFont);
            g.drawString(yLabel, (int)(0.005*xSize),(int)(0.06*ySize));
            g.drawString(xLabel, (int)(0.45*xSize),(int)(0.985*ySize));
            yAxisCoord=0;
            int xAxisCoord=0;
            if (yCase == 1)
            {     xAxisCoord=(int)(0.925*ySize);
      g.drawLine((int)(0.075*xSize),xAxisCoord,(int)(0.875*xSize),xAxis
Coord);
            }
            if (yOrderOfMag == 1)
            {     yTick = 10*yTick;
                  yOrderOfMag = 0;
            }

            yAxisCoord =  (int)(0.075*xSize);

            int TickCoord;
            int OOMCoord;
            if (yCase<3)
                  {     TickCoord = (xAxisCoord+(int)(0.03*ySize));
                        OOMCoord = (xAxisCoord+(int)(0.045*ySize));
                  }
                  else
                  {     TickCoord =(xAxisCoord-(int)(0.015*ySize));
                        OOMCoord = (xAxisCoord-(int)(0.03*ySize));
                  }
            if (xOrderOfMag != 1 && xOrderOfMag !=0)
            {     //g.drawString(("*10^" +
Integer.toString(xOrderOfMag)),(int)(0.91*xSize),OOMCoord);
            }


            else if (xOrderOfMag == 1)
            {     xTick = 10*xTick;
                  xOrderOfMag = 0;
```

```
            }

            if(tempo2>10)
            {
                  xTick=xTick/10;
                  xOrderOfMag = 0;
            }

            double Tick;
            double teste;
            switch (xCase)
            {     case 1:
                        {Tick = 0;
                        while
((Tick*Math.exp(xOrderOfMag*eValue))<(1.01*xDisplayMax))
                        {

                              xCoord =
(int)((((Tick*Math.exp(xOrderOfMag*eValue))/xDisplayRange)*0.8*xSize)+(
0.075*xSize));
                              g.drawLine(xCoord,(int)(xAxisCoord-
(0.005*ySize)),xCoord,(int)(xAxisCoord+(0.005*ySize)));

                              teste=(int)(Tick*100);

                              teste=teste/100;

                              Tick=(double)teste;


      g.drawString(Double.toString(Tick),(xCoord-
(int)(0.015*ySize)),TickCoord);

                              Tick = Tick + tempo2;

                        }
                        }
                        break;

                  }


            TickCoord = (int)(yAxisCoord-(0.07*ySize));

            switch (yCase)
            {     case 1:
                        {Tick = 0.0;
                        while ((Tick*Math.exp(yOrderOfMag*eValue)) <
(1.01*yDisplayMax))
                        {     yCoord = (int)(ySize -
((((Tick*Math.exp(yOrderOfMag*eValue))/yDisplayRange)*0.8*ySize)+(0.075
*ySize)));
                              g.drawLine((int)(yAxisCoord-
(0.005*xSize)),yCoord,(int)(yAxisCoord+(0.005*xSize)),yCoord);
```

```
      g.drawString(Double.toString(Tick),TickCoord,yCoord+(int)(0.015*y
Size));
                                    Tick = Tick + yTick;
                              }
                              }
                              break;

                        }


      }
}
```