UNIVERSIDADE DE SÃO PAULO
ESCOLA POLITÉCNICA

ALEX LOPES DE OLIVEIRA

Instrumentação inteligente via Web Services

São Paulo
2006

ALEX LOPES DE OLIVEIRA

Instrumentação inteligente via Web Services

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

Área de Concentração: Microeletrônica
Orientador: Prof. Dr. Francisco Javier Ramirez-Fernandez

São Paulo
2006

AUTORIZO A REPRODUÇÃO E DIVULGAÇÃO TOTAL OU PARCIAL DESTE TRABALHO, POR QUALQUER MEIO CONVENCIONAL OU ELETRÔNICO, PARA FINS DE ESTUDO E PESQUISA, DESDE QUE CITADA A FONTE.

Este exemplar foi revisado e alterado em relação à versão original, sob responsabilidade única do autor e com a anuência de seu orientador.

São Paulo, de de 2006

Assinatura do autor

Assinatura do orientador

FICHA CATALOGRÁFICA

Oliveira, Alex Lopes de
Instrumentação inteligente via Web Services / A.L. Oliveira. -- São Paulo, 2005.
p.97

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

1.Cliente/servidor 2.Sistemas de instrumentação elétrica 3.Internet 4.Sistemas embutidos I.Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia de Sistemas Eletrônicos II. t.

Dedico este trabalho à
minha mãe Áurea,
pelo seu carinho e dedicação,
à minha irmã Julianne,
ao meu sobrinho Luís Eduardo
e a memória do meu pai Francisco.

AGRADECIMENTOS

O meu principal agradecimento, sem sombra de dúvida, é ao Prof. Javier pois eu sei que o que eu sou hoje e o que eu conquistei não seria possível sem ele. Nestes oito anos, desde os tempos da graduação, ele sempre foi para mim um exemplo de dedicação, idealismo e ética (que tanto falta neste país). Quando vejo os novos alunos com as mesma dúvidas, incertezas e questionamentos que eu tinha no passado procuro ajudar com o melhor que eu tenho, e me surpreendo ensinando as mesmas coisas que em algum momento no passado o Prof. Javier ensinou a mim. As palavras escritas ficam na estante mas transmitir o que se recebeu é vida e é o melhor que posso fazer em agradecimento.

Dedico o meu agradecimento especial também a uma Professora que me ensinou a ler e a escrever, não somente a técnica mas algo muito mais difícil de se ensinar: ler e compreender o que é certo e errado, e escrever a verdade. O que era para ser difícil foi fácil, não necessitou de quadro e giz, bastou o seu amor, dedicação e compreensão, e o mais importante o exemplo de vida da Prof. Áurea, a minha Mãe.

O caminho é árduo mas as alegrias da convivência e o apoio dos amigos suavizam a jornada, os meus mais sinceros agradecimentos ao Prof. Walter Salcedo, Claudia, Michel, Elisabete Galeazzo, Maurício Perez, Mauro Braga, Ítalo Romani e Rodrigo Fazenda. Agradeço também as colaborações para este trabalho de André Nardelli, Jorge Corso, Luciano Ogiboski, Rafael Cassaniga, Valdo Reis e Thiago.

Serei sempre grato também aos inúmeros colaboradores anônimos da comunidade de desenvolvedores do Software Livre, que conseguem provar que a cooperação é a melhor solução técnica e humana em um mundo fragmentado pelo individualismo desumano. Ao CNPq pelo apoio financeiro.

A estrada continua sempre, e sempre,
A partir da porta onde se iniciou.
Agora a estrada já foi muito longe,
E eu devo seguir, se puder.
Prosseguindo nela com os meus pés cansados
Até que ela se una a uma via maior
Onde muitos caminhos e trajetos se encontram,
E para onde mais, não sei dizer.

J.R.R.Tolkien

RESUMO

Oliveira, A. L. **Instrumentação Inteligente via Web Services.** 2005. 97f. Dissertação (Mestrado) – Escola Politécnica da Universidade de São Paulo, São Paulo, 2005.

Neste trabalho é apresentado um sistema de instrumentação inteligente integrado com a arquitetura Web Services e desenvolvido em conformidade com as diretrizes estabelecidas na norma IEEE 1451.1. O protocolo de transmissão de dados do sistema é associado ao conceito das especificações dos transdutores e a um Web Service que permita a configuração remota do sistema de aquisição de dados. Através de uma interface de supervisão remota, acessível via Internet a partir de qualquer navegador, o usuário é capaz de identificar quais são os sensores inteligentes que estão conectados em uma rede, através de uma identificação única associada às suas características (tipo, formatação dos dados, etc.). Nesta mesma interface de apresentação o usuário tem a liberdade de selecionar o número de sensores que deseja ativar para efetuar algum monitoramento. Após a seleção dos sensores é disponibilizada uma interface de monitoramento que permite a visualização dos dados através de um gráfico bem como permite ao usuário salvar em um arquivo texto os dados coletados. Na mesma interface de monitoramento é disponibilizado o Web Service que permite alterar o intervalo de coleta de dados. Um conjunto de computadores em uma rede local (Intranet) simula a rede de sensores inteligentes. Nesta mesma rede está conectado o servidor responsável pela disponibilização da interface de supervisão remota.

Palavras-chave: Instrumentação Inteligente. Web Services. Redes de Sensores. Sensores Inteligentes.

ABSTRACT

Oliveira, A. L. **Instrumentação Inteligente via Web Services.** 2005. 97f. Dissertação (Mestrado) – Escola Politécnica da Universidade de São Paulo, São Paulo, 2005.

In this work is presented the integration of an established system of intelligent instrumentation with the Web Service's architecture, in accordance with the IEEE 1451.1 standard. It is suggested the association of a data-communication protocol with the concept of Transducer Electronic Data Sheet (TEDS) to a Web Service for making possible the remote configuration of a data acquisition system. Through a remote supervisor interface, accessible via Internet from any browser, the user can, at the first moment, identify which are the smart sensors connected in a network, using an unique identification associated to its characteristics (type, data formatting, etc.). In the same graphical interface the user can choose which sensors wants to monitor and, after the selection, will have available the monitoring interface that enables data visualization through a graph and allows the user to save the collected data in a text file. In the same monitoring interface the Web Service is also available; it admits modifications on the data collection interval. A group of computers in a local network (Intranet) simulates the smart sensors network. In the same network is connected a server responsible for supplying the remote supervisor interface.

Keywords: Intelligent Instrumentation. Web Services. Sensors Network. Smart Sensor.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

HTML	HyperText Markup Language.
HTTP	HyperText Transfer protocol.
IP	Internet Protocol.
NCAP	Network Capable Application Processor
STIM	Smart Transducer Interface Module
TCP	Transmission Control Protocol.
TEDS	Transducer Electronic Data Sheet
UDP	User Datagram Protocol.
XML	Extensible Markup Language
WWW	World Wide Web.

1. INTRODUÇÃO

Uma das principais dificuldades existentes em aplicações que envolvem redes de sensores é o gerenciamento da configuração da rede. Em uma rede de sensores, o sistema de controle central deve conhecer em tempo real quais são os tipos de sensores que estão conectados na rede, bem como quais novos sensores foram adicionados à rede. Em geral, a criação e manutenção desta configuração da rede de sensores são feitas manualmente. Um eventual problema nessa situação é relacionado à automatização deste processo onde uma possível solução pode ser implementada com o conceito de sensores inteligentes ou sensores *plug-and-play.* A especificação destes tipos de sensores é proposta pelas normas IEEE 1451.

A disponibilização de uma interface de monitoramento e controle na Internet de uma rede de sensores segue uma tendência consolidada já em outros sistemas de instrumentação, pois além de fornecer um acesso remoto utilizando uma infra-estrutura de rede já disseminada, permite a utilização de um conjunto de tecnologias associadas da World Wide Web (HTML, XML, SMTP, etc) que facilmente podem ser integradas a sistemas de informação já existentes (bancos de dados, sistemas de informação gerenciais, etc). Dentre estas tecnologias destaca-se a arquitetura Web Services, possibilitando a disponibilização das informações provenientes de rede de sensores como um *Web Service* , que pode ser solicitada através de uma mensagem seguindo o protocolo SOAP (Simple Object Access Protocol), com total independência das plataformas de desenvolvimento.

Conceitualmente, uma rede de sensores pode ser dividida em 3 estratos de natureza diferente e geralmente associados às funcionalidades ilustradas na Figura 1:

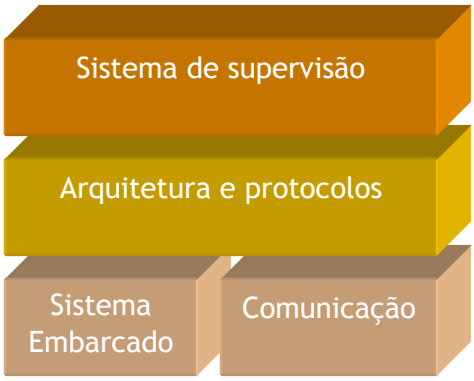

Figura 1. Diagrama de uma rede de sensores.

Sistema Embarcado: Responsável pela conversão analógica-digital dos sinais provenientes dos sensores emprega microcontroladores que devem ter confiabilidade, baixo consumo e custo, bem como memória suficiente não somente para o software embarcado mas também (no caso de sistemas de comunicação sem fio) para a pilha de comunicação.

Comunicação: As vantagens do emprego de um sistema de comunicação sem fio em uma rede de sensores, que a princípio espera-se que sejam disseminados em massa, podem ser inviabilizadas por questões de custo e consumo. Soluções novas como as tecnologias Bluetooth (FERRIGNO et al., 2005) e mais recentemente Zigbee (EGAN, 2005) atendem esta especificação, mas com um alcance curto (100m aproximadamente). A combinação destas tecnologias com tecnologias de longo alcance como as que empregam a rede de telefonia celular (GPRS e 1xRTT) pode ser uma solução por empregarem uma infra-estrutura amplamente disseminada, além do baixo custo (tarifação por volume de dados).

Arquitetura e protocolos: Em uma situação extrema, o arranjo físico dos sensores em uma rede não é conhecido previamente, o emprego de sistemas de comunicação de curta distância obriga o emprego de uma arquitetura *multi-hop*, na qual a comunicação entre os nós sensores é estabelecida entre eles para que a informação chegue ao concentrador, que além de agregar as informações, tem acesso à comunicação de longa distância. Estas

características particulares especificam por sua vez algumas particularidades que devem ser previstas nos protocolos de comunicação de uma rede de sensores:

- Protocolo plug-and-play de identificação do tipo do sensor, informação esta que deve estar associada fisicamente ao dispositivo;
- Rastreabilidade da informação, não basta transmitir os dados do sensor, mas é necessário identificar a fonte da informação;

Os protocolos de comunicação dos nós sensores em geral são bastante limitados, devido às restrições de processamento dos microcontroladores. Sendo assim, cabe ao concentrador, a conversão deste protocolo para um protocolo compreensível (em geral TCP/IP ou UDP) para o servidor, responsável pela recepção dos dados e disponibilização destes ao sistema de supervisão;

Sistema de supervisão: A recepção dos dados via conexão TCP/IP ou UDP deve ser pautada pela robustez do sistema em gerenciar e suportar diversas conexões simultaneamente, provenientes de diversos concentradores, assim como o registro destes dados em uma base de dados central. A interface de supervisão deve, além de informar os dados coletados, fornecer a localização da fonte dos dados, bem como os sensores disponíveis no momento. A associação da tecnologia Web Services permite a interoperabilidade de meios de acesso a estas informações, através da disponibilização das informações seguindo o protocolo SOAP, sendo transparente para o sistema, se a consulta está sendo feita por um computador pessoal ou por um celular.

2. SISTEMAS DE INSTRUMENTAÇÃO

Um sistema de instrumentação e medida para aquisição de dados pode ser definido como o conjunto de dispositivos que compõe uma cadeia pela qual os sinais físicos (de acordo com as suas características no tempo e na freqüência) podem ser medidos, graças à sua conversão para sinais elétricos e, posteriormente para o formato digital adequado à apresentação e processamento em um sistema computacional (LEE; SCHNEEMAN, 1999) (WUNNAYA; HOO, 1999). Obviamente, esta é uma definição moderna deste tipo de sistema, decorrente de uma série de avanços tecnológicos que diretamente ou indiretamente mudaram o conceito de instrumentação. Desses avanços podem-se destacar as tecnologias de conversão analógico-digitais, a popularização da presença dos computadores pessoais em sistemas de instrumentação, a Internet como interface de acesso remoto e a disseminação de sistemas embarcados dedicados à instrumentação, cada vez mais populares devido à redução dos custos dos dispositivos microeletrônicos e ao emprego de novas tecnologias de comunicação sem fio.

2.1. Sistemas de instrumentação associados a sistemas computacionais

No final da década de 60, a companhia americana Hewlett-Packard (nos dias de hoje a divisão de instrumentação desta empresa é conhecida como Agilent Technologies) projetou uma interface e um barramento de comunicação, para conectar seus instrumentos de medição a sistemas computacionais de controle e aquisição de dados, denominado HP-IB, um acrônimo de **H**ewlett **P**ackard **I**nterface **B**us. Esta interface rapidamente se popularizou devido à sua confiabilidade e velocidade (1 Mbps) o que contribuiu para que esta tecnologia fosse adotada como um padrão de interface de comunicação de equipamentos de instrumentação com sistemas computacionais, que permitiam a programação das ações a serem executadas por estes equipamentos (MARINO; NOGUEIRA; HERNANDEZ, 2000). A sua padronização foi estabelecida pelo IEEE em 1973. Desde então esta interface passou a ser conhecida como GPIB, acrônimo de **G**eneral **P**urpose **I**nterface **B**us, também conhecida pelo código IEEE 488.1, sendo que, em 1992, foi editada a segunda versão desta norma, que vigora até hoje, a

norma IEEE 488.2 (IEEE INSTRUMENTATION AND MEASUREMENT SOCIETY, 1992).

Uma arquitetura típica dos primórdios da interface HP-IB consistia no controle dos equipamentos de instrumentação através de sistemas computacionais bastante restritos, como exemplo ilustrativo, na década de 70 era bastante popular o emprego do HP 9825, considerado como um "*desktop computer*", pela sua fabricante a Hewlett-Packard, e não mais como uma calculadora, justamente por utilizar uma linguagem de programação o HPL, bastante semelhante ao BASIC. Esta linguagem de programação juntamente com uma placa de interface HP-IB permitia, através de um software, o controle de vários equipamentos de medição, conectados fisicamente através de um cabo apropriado a interface HP-IB (HICKS,2005).

Figura 2. Foto do equipamento HP 9825 associado a uma unidade de disco flexível. (HICKS, 2005).

Uma aplicação bastante comum deste sistema era a sua utilização em bancadas de testes de circuitos eletrônicos em indústrias, devido à repetição das rotinas de *set-up* dos equipamentos. É interessante observar que algumas indústrias nacionais utilizaram este sistema, inclusive com os mesmos equipamentos até a década de 90.

A evolução dos computadores pessoais, a partir da metade da década de 70, e das tecnologias de conversores analógico-digitais fez das placas de aquisição de dados (*DAQ*

boards) uma opção bastante atraente em sistemas de instrumentação, pela sua facilidade de implementação e versatilidade.

Dentro deste contexto surge um novo paradigma de sistemas de instrumentação baseado no conceito de Instrumentação Virtual, que pode ser definido como um sistema composto por um computador pessoal qualquer, um software dedicado à instrumentação, composto por uma interface gráfica de entrada / saída de dados que em geral procura ser semelhante à de um equipamento de instrumentação real, e placas de aquisição de dados com seus respectivos drivers (BORGES, 2002).

Destaca-se neste contexto a linguagem de programação gráfica LabVIEW, da companhia americana National Instruments, criado em 1986 originalmente para o microcomputador Macintosh hoje o LabVIEW é disponível para várias plataformas computacionais. Um software desenvolvido com o LabVIEW é composto por dois componentes um diagrama de blocos, que corresponderia as "linhas de código" do software, e o painel frontal, que permite a interação do usuário com o Instrumento Virtual (NATIONAL INSTRUMENTS, 2005). A vantagem de uma interface gráfica de programação está no fato de não ser necessário que o desenvolvedor do sistema de instrumentação tenha conhecimentos avançados de programação, que provavelmente seriam necessários caso fosse utilizada uma linguagem de programação genérica.

O principal concorrente do LabVIEW é a plataforma Agilent VEE (AGILENT TECHNOLOGIES, 2005), desenvolvido pela Agilent Technologies, e uma alternativa de software livre é a plataforma Comedi desenvolvida para o sistema operacional Linux (SCHLEEF; HESS, 2005).

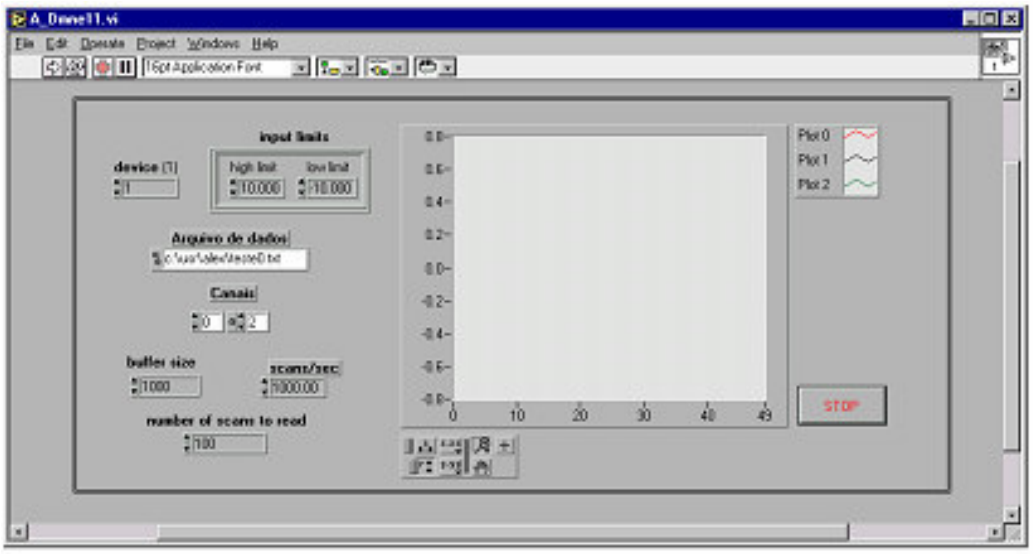

Figura 3. Painel frontal de um instrumento virtual desenvolvido com o LabVIEW (OLIVEIRA, 1999).

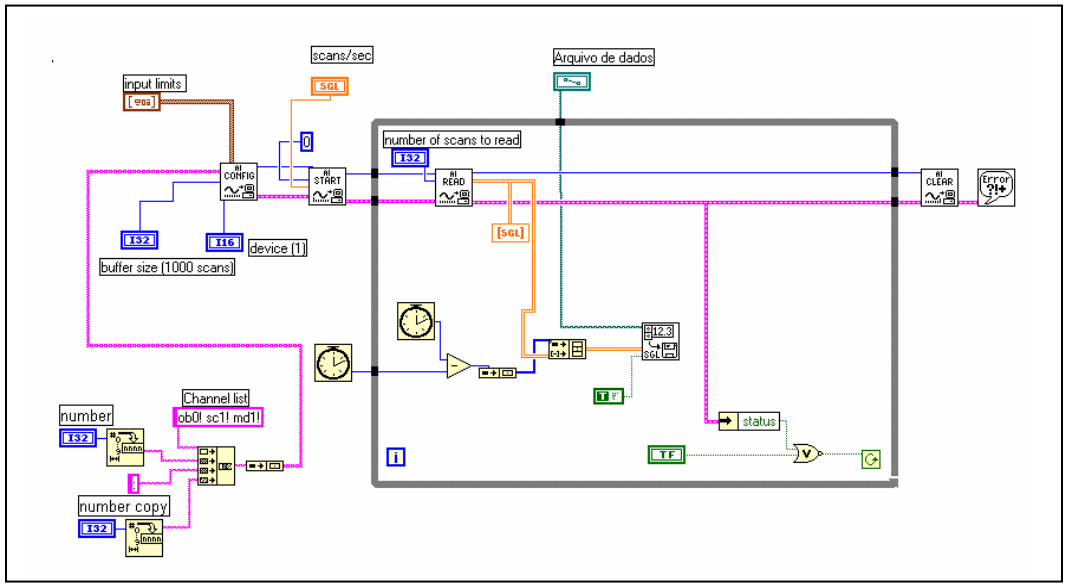

Figura 4. Diagrama de blocos de um instrumento virtual desenvolvido com o LabVIEW (OLIVEIRA, 1999).

2.2. Instrumentação remota via World Wide Web

A consolidação dos sistemas computacionais na área da instrumentação de imediato contribuiu de forma positiva em uma série de aspectos, tais como: armazenamento eletrônico dos dados, para posterior análise, automação do processo de controle e aquisição de dados, etc. Esses benefícios reforçaram esta consolidação o que contribuiu para que as inovações nas tecnologias computacionais produzissem evoluções nos próprios sistemas de instrumentação. Uma importante evolução que contribuiu de forma significativa foi a Internet.

Os serviços disponibilizados na Internet são serviços distribuídos, baseados principalmente no modelo cliente/servidor, no qual existem pelo menos dois componentes de software que interagem para fornecer um determinado serviço, sendo a World Wide Web o ambiente hipertextual no qual esses serviços são disponibilizados.

Serviços de disponibilização de textos, conteúdo multimídia e correio eletrônico são alguns dos serviços que rapidamente se popularizaram na Internet. A flexibilidade de acesso a informações através de uma rede mundial de computadores acessível através de um simples computador pessoal gerou novas possibilidades ao mundo da instrumentação, como a possibilidade de um sistema de instrumentação remota via Internet.

Diante desta nova perspectiva, em junho de 1999 o Instituto Nacional de Padrões e Tecnologia dos Estados Unidos (NIST) lançou a proposta de um *framework* para a padronização de sistemas distribuídos de controle e medição, baseado nas tecnologias disponibilizadas pela Internet. Nesse mesmo evento foi apresentado um sistema de controle e monitoramento remoto, via Internet, de um sistema de controle em malha fechada que regulava a temperatura de resfriamento da ferramenta de uma máquina industrial (LEE; SCHNEEMAN, 1999), como pode ser observado na figura 5.

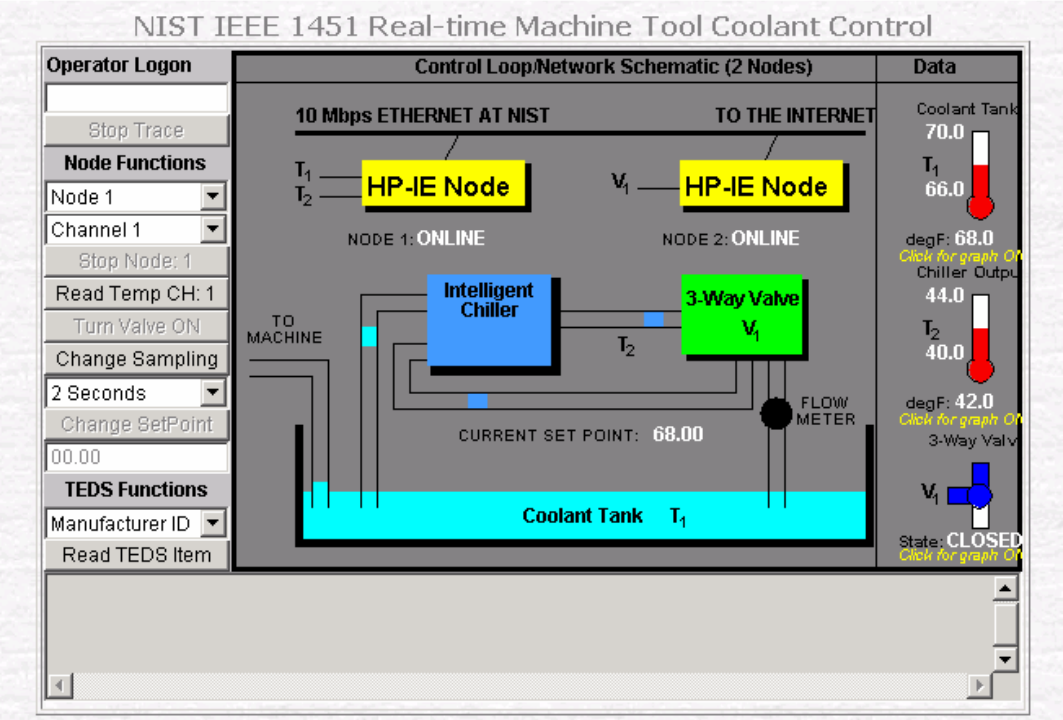

Figura 5 . Interface do sistema de controle e monitoramento remoto via Internet apresentada pelo NIST (NIST, 2005).

No mesmo período, ferramentas de Instrumentação Virtual como o LabVIEW e o HP-VEE começam a disponibilizar recursos para o desenvolvimento de sistemas de monitoramento e controle remoto. Inicialmente estes recursos possibilitavam a criação de um simples *socket* TCP/IP para a transmissão dos dados. Hoje em dia as últimas versões permitem a visualização dos painéis frontais dos instrumentos virtuais através de um browser qualquer.

Através do uso de tecnologias relacionadas com a Internet (Ethernet, TCP, HTTP, HTML, XML, etc) é possível controlar um dispositivo localizado em qualquer parte do mundo, transportar dados através de protocolos comuns garantindo a interoperabilidade entre diferentes dispositivos e, utilizar uma ampla infra-estrutura de rede já existente, sendo possível desenvolver sistemas de aquisição de dados e automação de sistemas sem necessidade de infra-estruturas adicionais (MCCLANAHAN, 2003) (PUPO, 2002).

No que se refere à interface homem-máquina, a utilização de um browser facilita significativamente o desenvolvimento desta interface, pois não é necessária a instalação de nenhum software cliente, dado que geralmente todos os computadores já têm um browser instalado; fácil interação, pois apresenta uma interface familiar ao operador e manutenção simples sendo somente necessário fazer as correções no lado do servidor.

2.3. Sistemas embarcados para instrumentação

Na década de 90 o pesquisador Mark Weiser, do centro de pesquisas da Xerox em Palo Alto, lança o conceito de Computação Ubíqua (também conhecida Computação Pervasiva). Neste conceito ele prevê uma nova tendência nos sistemas computacionais, inicialmente baseados em mainframes e posteriormente nos computadores pessoais, na qual os computadores estarão de tal forma embutidos no ambiente que os usuários terão uma interação com estes sistemas de uma maneira muito mais natural do que a atual. Uma das metas da Computação Ubíqua e habilitar dispositivos que permitam perceber as mudanças do ambiente e automaticamente se adaptarem e atuarem baseados nestas mudanças e nas necessidades e preferências do usuário. Esta tendência tem se tornado realidade graças ao avanço tecnológico nas mais diversas áreas tais como: computação distribuída, redes de sensores, interface homem-máquina, computação móvel, entre outras (WEISER, 1999).

Atualmente a grande maioria dos microprocessadores fabricados é usada em dispositivos que usualmente não são chamados de computadores. Eles estão presentes em toda parte: telefones celulares, caixas automáticos, veículos, fornos de microondas, aviões, equipamentos de automação industrial e comercial, equipamentos médicos, sistemas de segurança e em uma infinidade de outros equipamentos e dispositivos. Estes sistemas eletrônicos “dedicados” conhecidos como sistemas embarcados ou embutidos (do inglês *embedded systems*) são projetados para atender uma bem definida necessidade de um determinado momento através da associação de um hardware e um software, com um tempo de vida útil bem definido de alguns anos ou décadas e com possibilidades restritas de melhorias na mesma aplicação (CALVEZ, 1993). Além do mais estes

sistemas muitas vezes apresentam alguns pontos críticos no seu projeto, no que se refere ao consumo de energia, custo e confiabilidade.

Nos primórdios, os sistemas embarcados apresentavam uma série de restrições. Em geral empregavam um microprocessador de 8 bits de uso geral, que exigia uma memória adicional e o acréscimo de outros circuitos integrados periféricos, elevando o consumo de energia bem, como as dimensões físicas destes sistemas e o seu custo. Quanto ao software dos sistemas embarcados (*embedded software*) era originalmente desenvolvido de forma sintetizada para uma parte específica de um equipamento e empregava como linguagem de programação códigos de máquina, de baixo nível. Nos dias de hoje, é possível programar alguns microcontroladores utilizando inclusive linguagens de alto nível orientadas a objeto como C++ e Jstamp. Entre as vantagens dessa possibilidade pode-se destacar a criação de um hardware que pode solucionar uma família de problemas ao invés de se restringir a um problema específico, no qual a especialidade do hardware é dada pelo software, possibilitando um reuso do hardware.

Ao implementar um projeto de sistemas embarcados, existem algumas peculiaridades de projeto que devem ser seguidas, que dependendo do tipo de aplicação e finalidade do sistema são fundamentais para o êxito. Para isso alguns critérios podem ser seguidos:

- Lei de Moore - os sistemas embarcados estão cada vez mais complexos e sofisticados, necessitando de um software embarcado capaz de gerenciar tais recursos. A cada 18 meses, a capacidade de integração de circuitos eletrônicos dobra. Da mesma forma o software embarcado deve caminhar no mesmo ritmo de evolução dos circuitos integrados.
- Conectividade – aumenta cada vez mais a expectativa de que os sistemas embarcados tenham a capacidade de se interconectarem com outros equipamentos e dispositivos, utilizando meios e protocolos padronizados, tais como TCP/IP e tecnologias de comunicação sem fio, por exemplo.
- Ambiente Gráfico e Interface Homem-Máquina - cada vez mais são exigidas interfaces de maior complexidade e fáceis de utilizar, dependendo das particularidades do projeto.

- Custo de Licenciamento - projetos de sistemas embarcados geralmente prevêem uma fabricação em larga escala o que torna este fator muito importante, pois para cada equipamento ou dispositivo construído deverá ser adquirida uma licença de propriedade. Neste sentido, a utilização de um software embarcado baseado em um software livre leva uma grande vantagem.
- Disponibilidade do código fonte - sistemas embarcados via de regra necessitam de um software extremamente simplificado que atenda as suas funcionalidades específicas. A possibilidade do acesso ao código fonte é fundamental para a otimização das suas operações, o que torna a opção pelo software livre bastante vantajosa.
- Suporte a múltiplas arquiteturas de hardware - liberdade para escolha da plataforma de hardware, de modo que atenda aos requisitos do projeto, levando em conta aspectos como: custo, integração, consumo de energia e desempenho.
- Ferramentas de suporte ao desenvolvimento - disponibilidade de ferramentas para o desenvolvimento de aplicações, tais como, por exemplo, compiladores, depuradores de código, emuladores, etc.

Esses são alguns fatores que devem ser observados, levando-se em conta a aplicação do projeto envolvendo sistemas embarcados (REIS, 2002).

As perspectivas que a evolução dos sistemas embarcados propõem ao campo da instrumentação são inúmeras, principalmente no que se refere à possibilidade de se instalar sistemas de instrumentação em lugares de difícil acesso ou em ambientes críticos, que impedem a utilização de um computador pessoal. Um outro fator a favor do emprego de sistemas embarcados é o custo, menor que o de um computador pessoal ou de um equipamento completo de instrumentação, possibilitando o emprego de dispositivos de monitoramento e controle em massa.

Os sistemas embarcados voltados à instrumentação, em geral, são compostos por microcontroladores por apresentarem uma série de vantagens tais como a integração de várias funcionalidades em um único dispositivo, por exemplo: uma CPU de 8 ou 16 bits, um conversor analógico-digital com várias entradas multiplexadas, interface de comunicação serial RS-232, clock interno, que permite a geração de interrupções, geração de PWM bem como uma memória interna (GODFREY, 1996). Esta arquitetura

simplifica, em muito, o trabalho do projetista, no desenvolvimento de um sistema embarcado, voltado à instrumentação, facilitando a associação do microcontrolador a um conjunto de sensores e ao controle de atuadores. Outra opção é o emprego dos *Digital Signal Processors* (DSP), que, apesar de terem um custo mais elevado em relação aos microcontroladores (alguns modelos desses dispositivos, inclusive, já são disponibilizados por menos de US$ 1,00), são bastante recomendados em projetos que exigem uma alta velocidade de processamento de dados.

2.4. Tendências

Recentemente a evolução dos sistemas de comunicação tem contribuído para que surjam componentes e funções em vários sistemas embarcados, que permitem a substituição de soluções centralizadas, pela implementação de aplicações geograficamente distribuídas, conectadas entre si (OKAMOTO, 2000).

A integração de sistemas distribuídos, baseados no modelo cliente/servidor, em sistemas microcontrolados, tem sido acelerada pelo surgimento de novas tecnologias que facilitam essa integração como, por exemplo, microcontroladores que permitem uma conexão direta com uma rede Ethernet, permitindo inclusive a instalação de um software embarcado com instruções em HTML para a apresentação dos dados diretamente na Internet, com uma funcionalidade semelhante à de um servidor Web, graças à disponibilização de uma pilha TCP/IP no microcontrolador. Na figura 6 pode ser observado um exemplo de um módulo baseado em um microcontrolador que permite uma conexão direta com uma rede Ethernet a partir da disponibilização de uma pilha TCP/IP (RABBIT SEMICONDUCTOR, 2005)

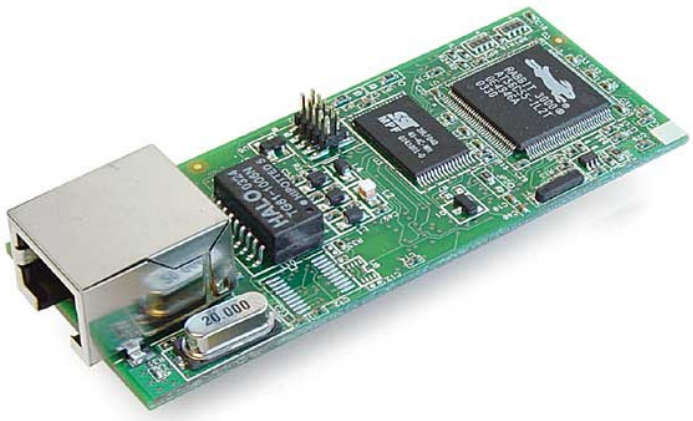

Figura 6. Exemplo de um módulo baseado em um microcontrolador com disponibilização de uma pilha TCP/IP.

Sistemas de comunicação sem fio estão cada vez mais integrados a sistemas embarcados, seja para transmissões de curta distância, em geral por volta de 100 metros, ou a grandes distâncias, via telefonia celular. É interessante frisar que, apesar de neste caso o meio físico ser diferente de uma rede Ethernet, em geral o software de gerenciamento da comunicação é muito semelhante, pois se utiliza das mesmas tecnologias de comunicação via Internet.

Existe a necessidade de que, cada vez mais, os sistemas embarcados sejam compactos e de baixo consumo, e, por outro lado, se exige cada vez mais que eles tenham a capacidade de executar operações cada vez mais sofisticadas. Estes fatos fazem com que o desempenho de um sistema autônomo microcontrolado seja um ponto crítico (TI-YEN, 1998) sendo que a conectividade pode constituir uma solução através de um processamento computacional distribuído.

2.5. Conclusões do capítulo

Como observado neste capítulo, a integração dos computadores no ambiente da instrumentação permitiu que os avanços e tendências da informática começassem a ter uma forte influência também na instrumentação. A Internet é um exemplo deste fenômeno. Paralelamente, os avanços da microeletrônica contribuíram para o surgimento de microcontroladores e sensores mais sofisticados, tornando convidativo o emprego desses dispositivos em sistemas de instrumentação. No momento atual há uma grande associação dessas duas tendências.

A Internet permite a integração de um número ilimitado de sistemas embarcados de instrumentação. Esta integração é estimulada ainda mais pelo baixo custo desses sistemas e pela facilidade de acesso remoto a esses dispositivos. Esta perspectiva se confronta por sua vez com alguns desafios:

- a diversidade de protocolos e plataformas de transmissão de dados na Internet dificulta a interoperabilidade dos sistemas. No caso de sistemas embarcados, a problemática pode ser mais crítica, como por exemplo em uma caso hipotético de migração de plataformas de supervisão que gere a necessidade de se alterar o software embarcado de cada um dos microcontroladores de um sistema de instrumentação;
- seguindo a tendência da disseminação em grande escala de sensores e atuadores associados a sistemas embarcados, conectados a uma rede de comunicações, espera-se que o sistema de supervisão seja capaz de identificar as alterações desta rede de instrumentação (mudança dos sensores/atuadores instalados, formatação específica dos dados, etc) e se auto-ajustar a estas mudanças, o que evitaria todo um gasto de tempo de reprogramação do sistema.

O presente trabalho contém análises e propostas que buscam soluções a estes desafios que atualmente são evidenciados mais intensamente em algumas aplicações de instrumentação, como é o caso das redes de sensores sem fio. É um caso de estudo bastante interessante, pois envolve todos os conceitos discutidos até o momento.

3. REDES DE SENSORES

Consideradas como uma das tecnologias mais promissoras do século XXI, as redes de sensores diferem das redes de computadores tradicionais em diversos aspectos. Em geral estas redes possuem um grande número de elementos distribuídos (sem um arranjo pré-definido, em algumas situações), têm restrições de consumo energético, e devem possuir mecanismos de auto-configuração e adaptação devido a problemas como falhas de comunicação. Uma rede de sensores tende a ser autônoma e requer um alto grau de cooperação para executar as tarefas definidas para a rede. Isto significa que algoritmos distribuídos tradicionais, como protocolos de comunicação e gerenciamento da configuração, devem ser revistos para esse tipo de ambiente antes de serem utilizados diretamente (CHONG; KUMAR, 2003) .

Espera-se que as soluções empregando redes de sensores se popularizem executando as tarefas mais diversas possíveis, sendo que elas têm o potencial de serem empregadas nas mais diversas áreas, tais como:

- **Meio ambiente:** Para monitorar variáveis climáticas (temperatura, umidade e pressão) bem como a presença e comportamento de determinadas espécies em um ecossistema. (SZEWCZYK, 2004).
- **Indústria:** Verificação em tempo real do bom funcionamento das máquinas de uma indústria, através do acompanhamento de algumas variáveis como nível de lubrificação, vibração, etc (CHONG; KUMAR, 2003).
- **Militar / Segurança:** Detecção de movimentos inimigos, explosões, a presença de armas químicas ou nucleares, etc. Neste tipo de aplicação, os requisitos de segurança são fundamentais no que se refere às transmissões dos sensores, sendo geralmente reduzidas para evitar interceptações, bem como criptografadas e submetidas a processos de assinatura digital (NEMEROFF, 2001).
- **Transportes:** Monitoramento do tráfego de veículos em rodovias, malhas viárias urbanas, etc (HSIEH, 2004).
- **Medicina:** Acompanhamento dos sinais vitais de um grande número de pacientes em tempo real, com a possibilidade de notificações de emergência a equipe

médica, bem como o registro destes dados no histórico do paciente para a posterior análise (MALAN, 2004).

Nas aplicações descritas acima, sensores devem ser conectados a outros sensores ou ainda a dispositivos de monitoramento, controle e aquisição de dados. Conectar esses sensores através de linhas de comunicação, como par trançado, cabo coaxial ou fibra óptica, é uma tarefa que pode não ser viável devido ao tipo de aplicação da rede, por exemplo, monitoramento ambiental em uma extensa e inóspita área geográfica, aplicações móveis de rastreamento de veículos ou devido à quantidade de sensores que devem ser interconectados. Além disso, uma característica da rede de sensores é a reconfiguração, isto significa que canais de comunicação que existiam podem terminar devido à destruição ou inatividade de sensores. O contrário também pode acontecer, ou seja, canais precisarem se tornar operacionais quando sensores presentes na rede ficarem ativos e novos sensores forem acrescentados. Logo, o custo de manutenção operacional de uma rede de sensores usando linhas de comunicação inclui o custo da própria linha física mais o custo de manutenção desse meio. Desta maneira a viabilização de uma rede de sensores depende fortemente da sua associação a tecnologias de comunicação sem fio (HILL, 2004).

3.1. Elementos de uma rede de sensores sem fio

Esta seção tem como objetivo descrever alguns dos principais elementos tecnológicos que formam uma rede de sensores sem fio, dentre os quais destacam-se:

- Nós sensores: a unidade básica de composição das redes de sensores;
- Nós gateway: responsável pela interface da rede de sensores com outras redes;
- Interfaces de comunicação sem fio: diversos avanços tecnológicos destes elementos têm contribuído para a viabilização das redes de sensores.

3.1.1. Nós sensores

Nós sensores são sistemas embarcados que em uma rede de sensores executam as tarefas de sensoriamento e atuação, processamento e comunicação, tendo como objeto de execução o fenômeno físico observado. Uma configuração típica do nó sensor é composta de cinco componentes: unidade de sensoriamento, conversor analógico-digital, unidade de processamento central, bateria e unidade de comunicação. Estes nós podem operar individualmente bem como formar uma rede com o objetivo de associar as informações individuais de cada sensor para monitorar algum fenômeno de forma cooperativa.

Os primeiros modelos de nós sensores apresentavam dimensões da ordem de dezenas de centímetros. Hoje já existem nós na ordem de milímetros, como é o caso do Smart Dust desenvolvido na Universidade de Berkeley na Califórnia (KAHN, 1999). A redução do tamanho do sensor, graças ao avanço da tecnologia de MEMS, tem contribuído para a redução dessas dimensões.

Um ponto crítico do projeto dos nós sensores é a autonomia energética. Há uma grande diferença entre as tecnologias de fabricação de baterias pois algumas são projetadas para serem utilizadas até o final de sua vida útil com uma única carga, enquanto outras prevêem várias recargas. A escolha da bateria a ser utilizada nos nós sensores deve considerar outras características, como volume, condições de temperatura e capacidade inicial (SOHRABI, 2000).

3.1.2. Nós de interface com outras redes

A comunicação da rede de sensores com outras redes ocorre através do nó gateway, conforme ilustrado na figura 8. Os dados coletados pelos nós sensores percorrem a rede de sensores até chegar a este nó que irá encaminhá-las, através de uma rede externa como a Internet, até o servidor da aplicação de monitoramento. Em termos de processamento, o nó gateway além de executar a conversão de protocolos, para permitir a interação entre as duas redes, também pode executar determinados algoritmos para descartar comunicações redundantes. Isto ocorre quando a mesma informação de um

nó sensor é recebida por caminhos diferentes dentro da rede de sensores. Além dessas funções este nó pode ser o consumidor final dos dados coletados para uma posterior coleta local, bem como receber as tarefas de sensoriamento da aplicação e difundir estas tarefas até que os nós sensores iniciem as suas atividades (YARVIS, 2002).

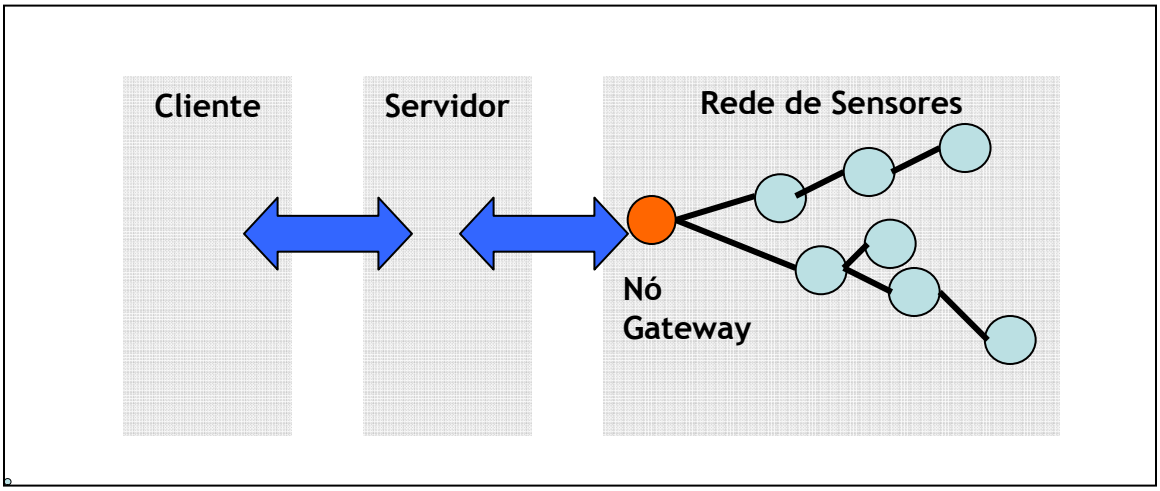

Figura 7. Arquitetura de uma rede de sensores associada a um nó gateway (YARVIS, 2002).

Em termos de implementação o nó gateway é um sistema embarcado semelhante ao nó sensor, mas requer alguns cuidados especiais em termos de projeto. Como ele é mais exigido em termos de comunicação e processamento de dados, o seu consumo energético é maior, necessitando de baterias com maior autonomia. A capacidade de processamento dos microcontroladores destes nós também deve ser maior.

3.1.3. Interfaces de comunicação sem fio

Entre as tecnologias de comunicação sem fio mais promissoras ao desenvolvimento de redes de sensores sem fio pode-se destacar: serviços de transmissão de dados via telefonia móvel (GPRS/1xRTT), a família IEEE 802.11, ZigBee, Bluetooth e RFID.

O padrão de comunicação para redes locais IEEE 802.11, também conhecido como Ethernet sem fio, foi proposto em 1995 com o objetivo de acrescentar uma nova

camada física ao modelo ISO/OSI visando estabelecer um padrão para redes locais sem fio (WLANs – *Wireless Local Networks*), para comunicação de dados com taxas de transferência de até 2 Mbps e prevendo a possibilidade de uso tanto de rádio freqüência quanto de infravermelho para a comunicação. Em 1999, o IEEE publicou duas extensões ao padrão, conhecidas por 802.11a e 802.11b, possibilitando taxas de transferência de até 54 Mbps e 11 Mbps respectivamente, usando exclusivamente rádio freqüência.

O 802.11a utiliza um esquema especial de multiplexação para atingir altas taxas de comunicação, o que torna impossível a comunicação entre dispositivos 802.11a e 802.11b. O padrão 802.11a pode ser utilizado em ambientes externos, pois a sua freqüência de operação de 5 GHz reduz consideravelmente a susceptibilidade a interferências.

O 802.11b opera na freqüência de 2,4 GHz conhecida como *Industrial, Scientific and Medical* (ISM). Esta faixa de freqüência é aberta para uso geral em um grande número de países, o que significa que cada dispositivo operando nesta faixa não necessita ser licenciado individualmente. Em 2003 foi lançada a extensão 802.11g , que tem se popularizado como uma solução substituta ao 802.11b, além de ser compatível com esta extensão também é com a 802.11a. A sua taxa de transmissão pode ser de 54 Mbps ou 11 Mbps, dependendo de qual compatibilidade for adotada. Atualmente já é comum existir uma infra-estrutura baseada no padrão 802.11 disponível para clientes em livrarias, cafés, e outros estabelecimentos comerciais (ONO, 2004).

Entre os serviços de transmissão de dados via telefonia móvel destacam-se a tecnologia *General Packet Radio Service* (GPRS), disponível aos usuários da tecnologia GSM. A tecnologia GPRS disponibiliza, na prática, uma capacidade de transmissão de dados de aproximadamente 40 kbps, enquanto a tecnologia 1xRTT oferece aproximadamente 100 kbps. A transmissão de dados no GPRS é feita através do envio de pacotes de dados, o que permite a transmissão simultânea de várias fontes em único canal, sendo que o canal de transmissão de dados é estabelecido nos intervalos de ociosidade do canal de voz. Esse recurso explica o baixo custo da tarifação da transmissão dos dados, que é feita pelo volume de dados transmitidos, em torno de US$ 2,00 por Megabyte. A conexão de dados pode seguir os mesmos protocolos usados na Internet (UDP, TCP, HTTP e SMTP) estabelecendo uma comunicação direta com um

servidor de recepção de dados através da Internet, uma opção bastante indicada no caso dos nós de interface (FERRER; OLIVER, 1998).

Uma das vantagens desses serviços de transmissão de dados é a infra-estrutura de comunicação, pois ela se utiliza das próprias estações de rádio-base da telefonia celular, o que é muito útil no caso de nós sensores que apresentam mobilidade, por outro lado existe a dependência da cobertura na região de emprego do sistema. Em geral tal cobertura é predominante nas áreas urbanas. Uma alternativa similar à tecnologia GPRS é a tecnologia 1xRTT, disponível aos usuários da tecnologia CDMA.

Para transmissões de dados a curta distância, aproximadamente 100 metros, duas tecnologias se destacam: a tecnologia Bluetooth e a Zigbee.

Em 1998 foi formado o Grupo de Interesse Bluetooth (*Bluetooth Special Interest Group*) para desenvolver uma tecnologia de comunicação sem fio baseada na norma IEEE 802.15.1 que fosse capaz de interligar aparelhos eletrônicos pessoais. Ele ainda permite a conexão entre diferentes tipos de dispositivos possibilitando a formação de redes ad-hoc. O Bluetooth tem uma arquitetura de comunicação definida em diversas camadas. O padrão Bluetooth 1.1 opera na faixa de freqüência de 2.4 GHz (ISM). Vários dispositivos Bluetooth podem se comunicar dentro de uma mesma área, a uma taxa de aproximadamente 1 Mbps.

A comunicação entre dois dispositivos Bluetooth é da forma mestre-escravo, e cada mestre pode se comunicar com até sete escravos ativos. Qualquer dispositivo pode ser mestre ou escravo, sendo que o seu papel é definido dinamicamente na conexão. O dispositivo que estabelece a conexão se torna o mestre. No entanto, os papéis podem ser trocados posteriormente. Um canal de comunicação compartilhado pelo mestre e pelos escravos é chamado piconet. Dentro de uma piconet, a comunicação se dá apenas entre o mestre e os escravos, não sendo permitida a comunicação entre escravos. Várias piconets dentro de uma mesma área de cobertura de sinal formam uma scatternet. O Bluetooth foi projetado de forma a permitir que várias piconets possam coexistir na mesma área, minimizando a interferência entre as redes (FERLINE, 2003)

A tecnologia Zigbee é uma tecnologia relativamente nova lançada em 2004 definida pela *ZigBee Alliance*, que é composta por várias empresas e coordenada pela Freescale, divisão de semicondutores da Motorola, baseada na norma 802.15.4 do IEEE,

e operando na banda ISM. Esta tecnologia, semelhante em termos de arquitetura de operação ao Bluetooth, contém algumas inovações que tornam o seu emprego promissor em redes de sensores.

Com uma taxa de transmissão de 250 kbps, suficiente para o seu emprego nos nós sensores (o que possibilita a sua comercialização a baixo custo viabilizando o seu emprego em massa), tem como principais diferenças em relação ao Bluetooth o tempo de acesso à rede formada pelos módulos, aproximadamente 100 vezes mais rápido, e um consumo energético, inferior ao do Bluetooth. Em aplicações Bluetooth, geralmente, os dispositivos são recarregados periodicamente, como celulares e PDA's, enquanto nos módulos Zigbee existe a possibilidade de estes serem alimentados com pilhas alcalinas comuns com autonomia de 2 anos (ZIGBEE ALLIANCE, 2005).

Tanto a tecnologia Bluetooth quanto a Zigbee apresentam algumas característica em comum, algumas delas bastante úteis na formação das redes de sensores, principalmente na camada dos nós sensores:

- redes formadas basicamente de dispositivos com baixo consumo de energia;
- as conexões entre dois dispositivos possuem diversos estados, com o objetivo de economizar energia e gerenciar a formação de outras redes;
- o estabelecimento da conexão entre dois elementos da rede passa por um procedimento de identificação e sincronização, que necessita de uma temporização para ocorrer efetivamente;
- formação espontânea de redes, possibilitando modificações constantes em sua topologia, sendo que essas modificações não são apenas em função da mobilidade como normalmente acontece nas redes sem fio;

Uma etiqueta de *Radio Frequency Identification* (RFID) é formada basicamente por uma antena, um capacitor e um minúsculo circuito integrado. Existem diferentes tipos de etiquetas para diferentes tipos de aplicações, sendo que o ponto importante para tornar a tecnologia RFID largamente aplicável é o seu baixo custo: cada etiqueta possui um identificador único que é enviado via difusão através da antena.

Etiquetas RFID podem ser ativas, passivas ou semi-passivas, e de leitura-escrita ou somente de leitura. Uma etiqueta RFID ativa tem uma bateria para a alimentação do circuito integrado e para enviar um sinal para uma estação de leitura. Uma etiqueta passiva não tem bateria e usa a própria energia das ondas eletromagnéticas enviadas pela estação de leitura para induzir uma corrente na antena da etiqueta que transmite o identificador. Etiquetas semi-passivas usam uma bateria para alimentar o circuito mas usam a energia eletromagnética para fazer a transmissão do identificador. Etiquetas de leitura-escrita podem gravar uma nova informação ou escrever sobre a existente enquanto uma etiqueta somente de leitura apenas transmite a informação gravada previamente. O raio de transmissão que um identificador alcança depende de fatores como potência do sinal transmitido e a freqüência de operação.

O RFID é uma tecnologia que pode ser utilizada em determinados casos particulares de redes de sensores, como em aplicações em que o nó sensor deve ser descartável, como por exemplo para o rastreamento de produtos podendo inclusive ser colado nas embalagens ou para a identificação de animais devido à sua leveza. O único porém é que os nós sensores devem ter a possibilidade de serem conduzidos às proximidades de uma estação de leitura. Resultados positivos desta tecnologia têm sido observados em aplicações de gerenciamento de cadeias de suprimentos (WANT, 2004).

3.2. Arquitetura de uma rede de sensores sem fio

Os elementos de uma rede móvel podem se comunicar diretamente (ponto-a-ponto) ou com o suporte de uma infra-estrutura. Redes que se comunicam da primeira forma são conhecidas como redes ad-hoc, também conhecidas como MANETs (*Mobile Ad hoc Networks*), e é o modelo de arquitetura que mais se assemelha com os modelos empregados em uma rede de sensores. Já as redes do segundo tipo são conhecidas como redes infra-estruturadas, e se utilizam de estações base para interconectar os dispositivos para prover o suporte à mobilidade. Neste caso os elementos móveis mesmo próximos uns dos outros estão impossibilitados de realizar qualquer tipo de comunicação direta, sendo que as estações de suporte à mobilidade podem estar conectadas a gateways que

permitem a comunicação entre os elementos móveis e a parte fixa da rede (MATEUS; LOUREIRO, 1998).

As redes de sensores sem fio podem ser vistas como um tipo especial de MANET, nas quais os elementos da rede trocam dados diretamente entre si. Do ponto de vista de organização, redes de sensores sem fio e redes móveis ad-hoc são idênticas, já que possuem uma topologia de rede que não é fixa e cada elemento da rede possui recursos energéticos escassos. No entanto MANETs têm como função básica prover um suporte à comunicação entre esses elementos computacionais, que individualmente, podem estar executando tarefas distintas (SOHRABI, 2000).

Os nós sensores, em geral, são organizados em grupos (*clusters*) onde pelo menos um dos nós sensores deve ser capaz de detectar um evento (por exemplo, mudança de temperatura, movimento, etc) na região, processá-lo e tomar uma decisão se deve fazer ou não uma difusão (*broadcast*) do resultado para outros nós. Nas redes organizadas em grupos estes podem ser homogêneos ou heterogêneos em relação aos tipos, dimensões e funcionalidades dos nós sensores. Nas redes homogêneas todos os nós sensores são idênticos em termos de autonomia das baterias, complexidade do hardware e funcionalidades, sendo que um dos nós é eleito líder do grupo, tornando-se, responsável pela agregação e transmissão dos dados, sendo que este nó pode ou não ser o próprio nó gateway. Quanto às redes heterogêneas são utilizados dois ou mais tipos diferentes de nós, com diferentes autonomias energéticas e funcionalidades (VIEIRA et al, 2003).

As redes de sensores também podem ser classificadas, quanto ao modo de roteamento, como *single-hop* e *multi-hop* . Na rede *single-hop* os nós sensores se comunicam direta e exclusivamente com o líder do grupo, enquanto que na *multi-hop* a comunicação entre os nós de um grupo é estabelecida para que a informação chegue ao líder do grupo.

Este aspecto da arquitetura de uma rede de sensores afeta diretamente o tempo de vida dos elemento de uma rede de sensores, que dependem da sua eficiência energética. Uma boa eficiência energética é apresentada pelas redes *multi-hop* com um rodízio do líder do grupo, para um balanceamento da carga, pois o nó líder tende a ter um consumo energético maior devido à comunicação com a estação base (JIANG; MANIVANNAN, 2004)

3.2.1. Especificação de uma rede de sensores sem fio

A especificação de uma rede de sensores sem fio deve levar em consideração alguns atributos que afetam diretamente a arquitetura desta rede. Entre estes atributos podem-se destacar:

Endereçamento dos sensores ou nós: Dependendo da aplicação, cada nó sensor pode ter um registro de identificação, único ou não. Por exemplo, sensores embutidos em peças numa linha de montagem ou colocados em animais devem ter um registro único caso seja necessário saber exatamente o local de onde o dado está sendo coletado. Por outro lado, nós sensores monitorando o ambiente numa região, possivelmente, não precisam ser identificados individualmente, pois a informação está associada ao valor de uma determinada variável referente à localização geográfica.

Síntese dos dados: Indica a capacidade de uma rede de sensores de sintetizar ou sumarizar dados coletados. Caso a rede tenha essa funcionalidade, é possível reduzir o número de mensagens que precisam ser transmitidas por ela.

Mobilidade dos sensores: Indica se os sensores podem se mover ou não em relação ao sistema em que estão coletando dados. Por exemplo, sensores colocados em uma estufa para coletar dados de umidade e temperatura são tipicamente estáticos, enquanto sensores colocados em uma bóia oceânica para medir o nível de poluição da água são móveis.

Latência: Especifica se os dados coletados pelos sensores têm algum tipo de restrição como um intervalo de tempo máximo para disseminação de seus valores para uma dada entidade de supervisão. Aplicações referentes à segurança apresentam justamente neste item o ponto mais crítico.

Limitação energética: Em muitas aplicações, os sensores serão colocados em áreas remotas, de difícil acesso a esses elementos para manutenção. Neste cenário, o tempo de

vida de um sensor depende da quantidade de energia disponível. Aplicações, protocolos, e algoritmos para redes de sensores sem fio não podem ser escolhidos considerando apenas sua funcionalidade e capacidade, mas devem levar em consideração a quantidade de energia consumida.

Tarefas colaborativas: O objetivo principal de uma rede de sensores sem fio é executar alguma tarefa colaborativa onde é importante detectar e estimar eventos de interesse e não apenas prover mecanismos de comunicação. Devido às restrições das redes de sensores sem fio, normalmente os dados são difundidos ou sumarizados para melhorar o desempenho no processo de detecção de eventos. O processo de sumarização é dependente da aplicação que está sendo executada (TILAK, 2002).

3.3. Conclusões do capítulo

Seja qual for a aplicação envolvida, a implantação de uma rede de sensores envolve atividades de disposição dos nós e formação da rede. Os nós sensores são geralmente lançados, sobre a área monitorada, de forma aleatória e despertam para a formação da rede. Antes de iniciarem as atividades de sensoriamento, os nós podem realizar atividades de descoberta, localização e de formação de grupos.

As redes de sensores são sistemas auto-organizados formados por nós sensores que podem espontaneamente criar uma rede não premeditada, agrupando-se e adaptando-se dinamicamente quando ocorrem falhas ou degradação de dispositivos, gerenciando o movimento dos nós e reagindo às trocas de tarefas e requisitos da rede. Os nós podem também se organizar para explorar a redundância resultante da alta densidade, assim como prolongar o tempo de vida do sistema.

Diversas pesquisas são conduzidas no âmbito de protocolos de roteamento para redes de sensores, alguns com ênfase na redução do consumo energético, outros na redução da latência na transmissão dos dados, ou no processo de localização dos nós, etc. Não é possível, segundo estudos, afirmar que um seja melhor que o outro. É uma solução de compromisso que depende da aplicação em questão (JIANG; MANIVANNAN, 2004). Porém, observa-se que há uma escassez de contribuições no sentido de integrar estes

conhecimentos referentes às arquiteturas e protocolos de uma rede de sensores ao conceito de sensores inteligentes, ou sensores *plug-and-play*, como é previsto pelas normas IEEE 1451.

4. INSTRUMENTAÇÃO INTELIGENTE

O conceito de instrumentação inteligente estava, no passado, associado a algumas funcionalidades executadas pelo próprio sistema de instrumentação embarcado como, por exemplo: linearização da resposta dos sensores, processamento de sinais e malhas de controle. A inovação destas implementações estava por conta das capacidades disponibilizadas pelos microcontroladores. Hoje este conceito remete principalmente à presença de sensores inteligentes no sistema de instrumentação.

Uma das principais dificuldades existentes em aplicações que envolvem redes de sensores é o gerenciamento da configuração da rede, principalmente de uma rede de sensores sem fio. O sistema de controle central deve saber, em tempo real, quais são os sensores que estão conectados na rede, sob qual faixa de comunicação sem fio está sendo estabelecida a conexão, bem como quais novos sensores foram agregados à rede. Em geral, a criação e manutenção desta configuração da rede de sensores é feita manualmente. Uma possível solução para a automatização deste processo é o conceito de sensores inteligentes ou sensores *plug-and-play* , pois:

- Uma arquitetura com uma interface elétrica com a rede, padronizada, permitindo que uma ampla variedade de tipos de sensores seja usada na mesma rede;
- Um protocolo de auto-identificação permitiria que a rede se configurasse dinamicamente.

Atualmente, os fabricantes competem por disponibilizar sensores cada vez mais inteligentes, mais baratos, compatíveis com o maior número de protocolos de rede e auxiliados pelas melhores ferramentas de software. Essa progressiva digitalização dos sistemas de medida e controle é vantajosa por reduzir a fiação analógica e facilitar a instalação, manutenção e expansão do próprio sistema.

No entanto a proliferação de inúmeros protocolos de rede, cada um com as suas vantagens e desvantagens, dificulta a integração de produtos de diferentes fabricantes e direciona o mercado para soluções proprietárias, normalmente mais caras e menos flexíveis. Uma possível solução deste problema é adotar uma interface universal para transdutores inteligentes, isto é, sensores e atuadores inteligentes, compatível com todos

os protocolos de rede, e que, em última análise, permita a integração automática de qualquer transdutor em qualquer rede. É neste contexto que são propostas as normas IEEE 1451.

4.1. Normas IEEE 1451

As normas IEEE 1451 não propõem um novo protocolo de rede. Em vez disso, elas definem um conjunto de interfaces normalizadas de hardware e software no sentido de separar o projeto dos transdutores da escolha das redes de comunicação. Neste sentido está o fato do modelo não impor quaisquer restrições aos aspectos construtivos dos transdutores, nem aos aspectos protocolares das redes. Ao separar as duas entidades, transdutores e redes, o modelo permite que o fabricante de transdutores se concentre no dispositivo de transdução, sem ter de se preocupar em adaptá-lo às muitas redes já existentes. Isso contribuirá para a melhoria da qualidade dos transdutores e para a redução do seu custo. O objetivo final do modelo IEEE 1451 é permitir o funcionamento imediato de qualquer transdutor em qualquer rede (KANG; SONG, 2004).

4.1.1. Histórico

Em setembro de 1993 se iniciaram as atividades de desenvolvimento das normas, promovidas pelas instituições National Institute of Standards and Technology (NIST) e Institute of Electrical and Electronic Engineers (IEEE). Na primeira reunião de trabalho foi percebida a necessidade de se criar uma interface comum de comunicação para transdutores inteligentes. Para tal atividade foram organizados os seguintes grupos de trabalho, atuantes nos diversos aspectos dessa interface (NIST, 2005) :

P1451.0 : proposto para definir um conjunto de APIs (A*pplication Programming Interfaces*), independentes da forma de implementação, que contenha uma série de funcionalidades para sistemas baseados nas normas IEEE 1451, para controle e gerenciamento de transdutores inteligentes;

P1451.1 : criado para definir a arquitetura de um modelo que estabelecesse uma interface entre um transdutor inteligente e uma rede qualquer de comunicação. Este modelo ficou conhecido como *Network Capable Application Processor* (NCAP);

P1451.2 : constituído para definir um padrão de referência para o módulo de transdução inteligente, em inglês *Smart Transducer Interface Module* (STIM),. Este modelo descreve as funcionalidades do transdutor, a sua interface digital de comunicação com o NCAP e a sua memória descritiva, conhecida como *Transducer Electronic Data Sheet* (TEDS);

P1451.3 : para o qual foi atribuída a tarefa de definir uma interface de comunicação digital para sistemas distribuídos de transdutores inteligentes;

P1451.4 : teve como objetivo proposto a definição de uma interface de comunicação dentro do conceito de sensores *plug and play* para transdutores com saída analógica;

P1451.5 : busca definir um conjunto de padrões que defina os métodos e formatos de dados para a comunicação sem fio entre transdutores que sigam as normas IEEE 1451;

P1451.6 : proposto para estabelecer uma interface entre uma rede baseada no protocolo CANopen e módulos de transdutores multicanais padronizados pelas normas IEEE 1451.

Cronologicamente os trabalhos evoluíram da seguinte forma:

- Março de 1993: surge a idéia de criar uma interface de comunicação para transdutores inteligentes;
- Março de 1994: primeira conferência promovida pelo NIST e IEEE para normalizar a interface;
- Setembro de 1994: primeira demonstração de um TEDS na Sensors Expo de Cleveland e criação dos grupos de trabalho P1451.1 e P1451.2;

- Março de 1995: apresentação da idéia de STIM e de NCAP na feira Sensors Expo de Boston, também marcada pela demonstração de sensores *plug & play*;
- Novembro de 1995: demonstração de um modelo de programação orientado a objeto para representar um transdutor inteligente genérico (norma P1451.1);
- Agosto de 1996: a versão provisória da norma P1451.2 passa a ser analisada no IEEE;
- Setembro de 1996: criação dos grupos de trabalho P1451.3 e P1451.4;
- Setembro de 1997: o IEEE adota a norma IEEE Std 1451.2-1997;
- Setembro de 1999: o IEEE adota a norma IEEE Std 1451.1-1999;
- Junho de 2002: criação do grupo de trabalho P1451.5;
- Março de 2003: criação do grupo de trabalho P1451.0;
- Setembro de 2003: o IEEE adota a norma IEEE Std 1451.3-2003;
- Março de 2004: criação do grupo de trabalho P1451.6;
- Maio de 2004: o IEEE adota a norma IEEE Std 1451.4-2004;

4.1.2. Norma IEEE 1451.1

A norma IEEE 1451.1 especifica, através de um modelo orientado a objeto, a arquitetura do software que gerencia a comunicação de um sensor inteligente com uma rede de acesso remoto aos dados. A especificação abstrai detalhamentos referentes ao "como" a arquitetura deve ser implementada, isto é, independe da tecnologia empregada. Esta especificação é composta por três definições fundamentais: o *Network Capable Application Processor* (NCAP), que estabelece a modelagem propriamente dita do sensor inteligente conectado a uma rede; o modelo de dados, que define a formatação dos dados fornecidos pelo sensor; e o modelo de comunicação, que especifica as arquiteturas de comunicação com a rede permitidas (IEEE, 2000).

O NCAP engloba um conjunto de classes, métodos e atributos que podem ser agrupados em quatro grandes blocos (SVEDA, 2004):

Classes: O bloco central do NCAP que providencia um ambiente de execução multitarefa independente, assegurando serviços básicos, tais como o

gerenciamento da memória, o tratamento de exceções e o gerenciamento de tarefas, o que inclui o agendamento e a execução das tarefas concorrentes, bem como mecanismos de comunicação e sincronização entre elas.

Funções: Os blocos de funções previstos na Norma 1451.1 constituem uma estrutura básica com a qual o desenvolvedor pode criar aplicações específicas. Estas estruturas não especificam "como" a aplicação deve ser desenvolvida, mas orientam o seu desenvolvimento através da listagem dos parâmetros empregados no sistema para que a aplicação seja o mais transparente possível para o NCAP.

Transdutor: O bloco de interface com o transdutor gerencia a comunicação entre o NCAP e o transdutor. Este processo se inicia com a identificação do tipo do transdutor que está conectado ao sistema, posteriormente com a disponibilização dos métodos de leitura e escrita dos transdutores identificados.

Rede: O bloco de interface com a rede trata de todas as transações entre o NCAP e a própria rede. A interação entre o NCAP e a rede baseia-se no conceito de chamada remota de métodos (RPC - *Remote Procedure Call*), permitindo a um cliente remoto, de forma absolutamente transparente, instanciar uma classe do NCAP e utilizar os seus atributos, métodos e eventos como se o objeto residisse no próprio cliente, adaptando desta forma, o conceito de serviço remoto para os transdutores.

4.1.3. Norma IEEE 1451.2

A norma IEEE 1451.2 define a arquitetura do módulo de transdução inteligente (STIM – *Smart Transducer Interface Module*) e a conexão deste ao NCAP. No STIM estão conectados todos os sensores e atuadores do sistema, digitais ou analógicos, responsáveis pela interação com o meio físico. Eventualmente, ele também é responsável pela aquisição, condicionamento e digitalização dos sinais analógicos provenientes de sensores analógicos. Neste aspecto a norma 1451.2 não impõe quaisquer restrições, tendo

o desenvolvedor liberdade para escolher os dispositivos que forem melhor convenientes. O que a norma regulamenta é a interação entre o módulo de transdução inteligente e o NCAP (CONWAY, 2000).

Do ponto de vista funcional o NCAP acessa o STIM como se este fosse uma extensão da memória, sendo que todas as funções do STIM são requisitadas através de um endereço de função e são encaminhadas para um sensor ou atuador através de um endereço de canal, no vocabulário da norma um canal representa um transdutor do STIM. O canal 0 tem a particularidade de representar todo o STIM, servindo para difundir e armazenar informação comum a todos os canais.

Também é definida pela norma a interface física, bem como o protocolo de comunicação digital entre o STIM e o NCAP, denominada interface independente do transdutor, em inglês *Transducer Independent Interface* (TII). Esta interface é composta de um barramento de dados, de 10 vias, e especifica todo o protocolo de transporte de dados.

A norma IEEE 1451.2 define uma memória descritiva do STIM cuja designação original em inglês é *Transducer Electronic Data Sheet* (TEDS). O TEDS é fundamental para a adição de novos transdutores ao sistema, pois quando um novo STIM se conecta ao NCAP, o seu TEDS é imediatamente lido de modo que o NCAP e a rede ficam automaticamente cientes das suas características, principalmente no que se refere aos tipos de transdutores associados ao STIM (IEEE, 1998).

A norma 1451.2 prevê oito estruturas TEDS que deverão ser implementadas em memória não volátil, das quais duas são obrigatórias e seis são opcionais. Distinguem-se estruturas em formato de texto e outras em formato binário. Algumas das estruturas têm direitos de leitura e de escrita, outras só podem ser lidas. Segue-se uma breve descrição de cada uma delas (VIEGAS, 2003):

Meta TEDS (estrutura obrigatória, formato binário, somente leitura) : O meta TEDS é uma estrutura de memória que contém as características gerais do STIM, tais como, identificador único, número de canais implementados e velocidade máxima de comunicação com o NCAP;

TEDS de Canal (estrutura obrigatória, formato binário, somente leitura): O TEDS de canal é uma estrutura de memória que contém as características particulares de cada canal do STIM, tais como tipo de canal, unidades físicas, valores máximo e mínimo, restrições temporais e precisão. Existe um TEDS de canal para cada canal do STIM, exceto para o canal 0 que é descrito pelo Meta TEDS.

TEDS de Calibração (estrutura opcional, formato binário, leitura e escrita): No caso de algum canal do STIM necessitar de correção por software, será associado um TEDS de calibração contendo os coeficientes da equação de correção. Além disso, o TEDS de calibração disponibiliza informação relativa à data da última calibração e ao período de calibração.

Meta TEDS de Identificação (estrutura opcional, formato de texto, somente leitura): O meta TEDS de identificação é um estrutura de memória que identifica detalhadamente o STIM. Nele estão previstos campos de texto descrevendo o nome do fabricante, modelo, número de série, códigos de revisão e descrição geral do produto.

TEDS de Identificação de Canal (estrutura opcional, formato de texto, somente leitura): Estrutura TEDS idêntica à anterior mas repetida para cada canal do STIM , com exceção do canal 0.

TEDS de Identificação de Calibração (estrutura opcional, formato de texto, leitura e escrita) : Toda a informação relevante ao processo de calibração do STIM, não prevista no TEDS de calibração, está disponível nesta estrutura de memória.

TEDS do Usuário (estrutura opcional, formato de texto, leitura e escrita): Estrutura de memória onde reside toda a informação livre que o usuário deseja incluir no STIM.

TEDS para Extensão: Estrutura de memória opcional, com formato e direitos indefinidos, reservada para extensões futuras da norma 1451.2.

4.1.4. Norma IEEE 1451.3

A norma IEEE 1451.3 propõe uma interface normalizada para sistemas de transdutores inteligentes distribuídos, através da definição de estruturas de memória TEDS, protocolos de identificação e sincronização de canais e funções de escrita e de leitura do transdutor. Tal como na norma IEEE 1451.2 detalhamentos de implementação não são especificados na norma. A interface proposta associa em um barramento comum o NCAP e vários módulos de transdução inteligente, nomeados como *Transducer Bus Interface Module* (TBIM) (IEEE, 2004).

O barramento é controlado por um mestre integrado no NCAP, podendo suportar vários módulos de transdução inteligente. Cada módulo pode suportar até cinco canais de comunicação multiplexados em freqüência sobre a linha de transmissão, permitindo que a alimentação dos módulos e a comunicação coexistam na mesma linha. Dos cinco canais de comunicação, apenas os canais de rede e de controle são obrigatórios.

As estruturas de memória TEDS propostas pela norma 1451.3 são uma variante das estruturas apresentadas para a norma 1451.2. Estão previstas duas estruturas de memória obrigatórias, em formato binário, denominadas meta TEDS de módulo e TEDS específico do transdutor. A primeira contém as características gerais do TBIM e a segunda descreve as características particulares de um canal do TBIM. Cada TBIM contém um meta TEDS de módulo e vários TEDS específicos do transdutor, um para cada canal de transdução (VIEGAS, 2003).

4.1.5. Norma IEEE 1451.4

A norma 1451.4 propõe a padronização de uma interface de comunicação para transdutores analógicos com o objetivo de compatibilizar os transdutores já existentes no

mercado (a maioria deles com interface analógica em tensão ou corrente) com as normas IEEE 1451.

A proposta consiste em sobrepor ou adicionar à interface analógica do transdutor, uma interface digital, de baixo custo, que acesse uma memória TEDS e permita mecanismos de auto-identificação e auto-calibração do transdutor, resultando em uma interface de modo misto, *Mixed Mode Interface* (MMI) (IEEE, 2005).

No que se refere à interface digital ela foi baseada no protocolo 1-Wire desenvolvido pela empresa Maxim / Dallas Semiconductor. Trata-se de um protocolo de comunicação serial do tipo mestre/escravo, suportado numa única linha com lógica positiva e baseado em largura de pulso.

As estruturas de memória TEDS propostas pela norma 1451.4 são um subconjunto das estruturas contempladas na norma 1451.2. Estão previstos campos para identificação do fabricante, modelo, número de série, incerteza, coeficientes de calibração, entre outros itens

4.2. Conclusões do capítulo

As Normas IEEE 1451 como um todo têm a preocupação de sempre manterem uma estrutura de módulos independentes, por exemplo, para o NCAP é indiferente a sua associação a um STIM (1451.2) ou a um TBIM (1451.3). Esta independência também é mantida no que se refere ao tipo dos dados que circulam entre estes módulos.

Existem vários exemplos de implementação destas normas em redes estruturadas, tais como CAN e Ethernet, por sua vez a implementação em redes não-estruturadas como no caso das redes de sensores sem fio não é trivial (LEE et al., 2004) (JOUVRAY et al., 2004) (AL-ALI et al., 2005). A dificuldade surge na necessidade de um protocolo de roteamento, que pode ter a sua complexidade aumentada no caso de redes *multi-hop* . Neste cenário complexo uma possível estratégia é o emprego de uma nova camada de comunicação, que deve ser fortemente associada aos dados fornecidos pelos sensores, para que possa ser utilizado como um meio de informações sobre o roteamento dos dados, bem como sobre os próprios dados. Esta camada de comunicação pode ser exercida pelo emprego de metadados, como é o caso da linguagem XML (*meta-markup*

language) e de soluções que empregam esta linguagem como é o caso da tecnologia Web Services.

5. WEB SERVICES

5.1. Arquitetura

Web Services pode ser definida como uma tecnologia de sistemas distribuídos com uma arquitetura orientada a serviço (*Service-Oriented Architecture* – SOA), composta de módulos de software, independentes e auto-descritos, o que permite que eles sejam identificados e requisitados por outros sistemas através de uma Internet ou Intranet. Um serviço nesta conceituação é definido como uma representação funcional de uma atividade do mundo real, com uma interface auto-descrita que independe de qualquer linguagem de programação específica (VOGELS, 2003).

Entre as principais características dos Web Services destaca-se (ARRUDA JR., 2003):

- Disponibilidade: os serviços fornecidos devem estar disponíveis aos sistemas de software que desejem utilizá-los;
- Descrição: devem possuir um descrição possível de ser entendida pelos computadores e utilizada para identificar a localização dos serviços e o modo como podem ser acessados;
- Interoperabilidade: definem um padrão comum que permite que diferentes sistemas atuem em conjunto, de maneira que, por exemplo, uma aplicação desenvolvida em C possa acessar serviços implementados em Java;
- Padronização: baseia-se em padrões já estabelecidos e gerenciados pela World Wide Web Consortium (W3C), entidade que através do desenvolvimento de protocolos comuns procura contribuir para o desenvolvimento da World Wide Web garantindo a sua Interoperabilidade;
- Propósito geral: podem prover infra-estrutura para múltiplos propósitos podendo ser especializada para cada tipo de sistema de software.

Uma SOA é estabelecida por três elementos que a definem (W3C, 2004a):

- *Service Provider*: na perspectiva da arquitetura SOA é a plataforma que hospeda o acesso ao serviço;
- *Service Requester:* é a aplicação que está procurando, requisitando e iniciando a interação com um serviço;
- *Service Registry:* é um repositório onde são registradas e publicadas pelo *Service Provider* as descrições dos serviços. E é através dele que o *Service Requester* encontra os serviços requisitados.

Por sua vez, a interação entre estes elementos é regida por um conjunto de operações (W3C, 2004a):

- *Publish*: para ser acessível, um serviço necessita ser descrito e publicado no *Service Registry,* mas antes de ser publicado o *Service Provider* necessita obter uma autenticação do *Service Registry* ,para publicar e modificar a descrição do serviço;
- *Discover*: após a operação de publicação o *Service Registry* necessita fornecer uma interface específica para receber a requisição do *Service Requester,* na operação de descoberta o *Service Requester* recupera diretamente a descrição do serviço ou faz uma solicitação ao *Service Registry* baseada no tipo do serviço requisitado.
- *Bind*: nesta operação o *Service Requester* acessa a informação de conexão no *Service Registry,* podendo desta maneira obter detalhes sobre os requisitos de chamada do serviço através da análise da informação, incluindo o caminho de acesso ao serviço, os parâmetros de chamada, os valores de retorno, protocolo de transferência e requisitos de segurança. Baseado nestas informações o *Service Requester* pode configurar a aplicação e implementar a chamada remota do serviço.

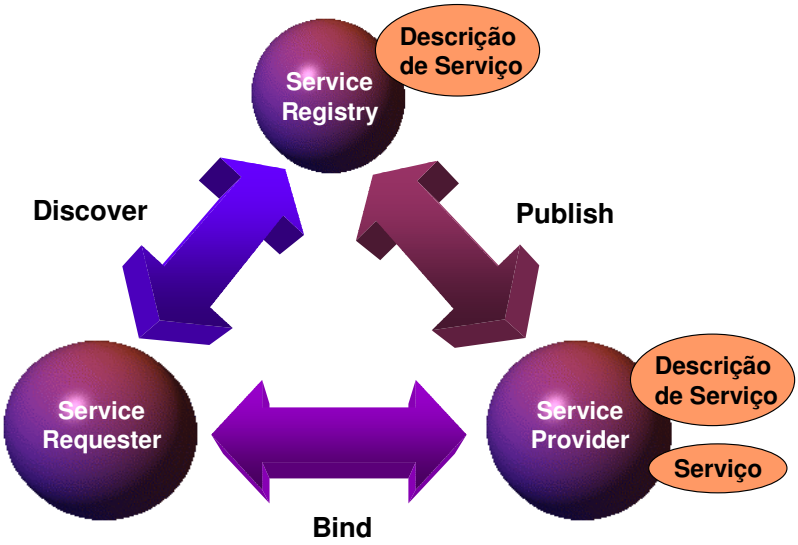

Figura 8. Representação esquemática de uma arquitetura orientada a serviço.

Para o atendimento dessas operações a tecnologia Web Services propõe um padrão de arquitetura baseada em camadas, com serviços e inter-relacionamentos estabelecidos por um conjunto de protocolos e linguagens. Nos fundamentos desse conjunto de camadas estão os serviços de transporte da rede, promovido como por exemplo pelo *Hypertext Transfer Protocol* (HTTP), amplamente utilizado em aplicações Web Services voltadas à Internet, mas não se restringindo a este. Outros protocolos podem ser usados nesta camada, como o *File Transfer Protocol* (FTP), o *Simple Mail Transfer Protocol* (SMTP) entre outros.

Camada de serviço	Descrição
Transport	Responsável pelo transporte de mensagens entre os aplicativos. No caso particular do Web Service podem ser utilizadas as vantagens do HTTP, que é o mesmo protocolo utilizado em aplicações utilizando a tecnologia WWW.
Information	Esta camada tem um papel fundamental na troca de dados e na padronização da semântica empregada. Um formato padrão amplamente aceito é o *eXtensible Markup Language (XML).*
Packaging	Nesta camada é estruturada a troca de mensagens e a comunicação entre aplicações. *Simple Object Access Protocol (SOAP)* é o protocolo consolidado nesta camada.
Description	Responsável por descrever a interface pública para um Web Service especifico. Esta camada é manipulada via WSDL (Web Services Description Language)

Tabela 1. Camadas de serviço e seus respectivos protocolos que compõem arquitetura da tecnologia Web Services.

Nas camadas superiores se encontram os protocolos e linguagens que formam a identidade da tecnologia Web Services como o protocolo *Simple Object Access Protocol* (SOAP), baseado na linguagem *eXtensible Markup Language* (XML), a linguagem de descrição *Web Services Description Language* (WSDL).

5.2. Linguagem XML

É de fundamental importância no Web Services o papel da linguagem (XML) e sua integração ao protocolo SOAP. O XML é uma linguagem de marcação de dados (*meta-markup language*) que provê um formato para descrever dados estruturados. Isso facilita

declarações mais precisas do conteúdo e resultados mais significativos de busca através de múltiplas plataformas. O XML permite a definição de um número infinito de *tags*. Enquanto que no conhecido *HyperText Markup Language* (HTML), as *tags* podem ser usadas para definir a formatação de caracteres e parágrafos, o XML provê um sistema para criar *tags* para dados estruturados. Um elemento XML pode ter dados declarados como sendo temperatura ambiente, pressão, a quantidade de óleo, ou qualquer outro tipo de elemento de dado. Como as *tags* XML são adotadas por intranets de organizações, e também via Internet, haverá uma correspondente habilidade em manipular e procurar dados independentemente das aplicações onde os mesmos são encontrados. Uma vez que o dado foi encontrado, ele pode ser distribuído pela rede e apresentado em um *browser* de várias formas possíveis, ou então esse dado pode ser transferido para outras aplicações, para processamento futuro e visualização. Devido ao fato do formato do documento ser aberto e flexível, ele pode ser usado em qualquer lugar onde a troca ou transferência de informação for necessária (W3C, 2004b).

As implementações que utilizam XML necessitam normalmente de um mecanismo de validação (Parser XML) para assegurar que o documento foi corretamente estruturado em relação a uma gramática adotada. Exemplos de gramáticas são o *Document Type Definition* (DTD) e o XML Schema. O XML Schema é a gramática recomendada pelo W3C por ter sido desenvolvida com o intuito de superar algumas limitações do DTD, tais como (ARRUDA JR., 2003):

- comandos e construtores que são difíceis de serem lidos, entendidos e mantidos pelos usuários finais;
- são especificados para estruturar dados XML, mas não são escritos em XML ao contrário do XML Schema.

5.3. Protocolo SOAP

SOAP é um protocolo baseado no XML para a troca estruturada de dados entre aplicações em rede (CURBERA, 2002). O SOAP consiste de três partes: um envelope que define um *framework* para descrição do que está na mensagem, um conjunto de regras de codificação para definir o conteúdo da aplicação (*header*) e uma convenção

para representar as chamadas e respostas de procedimento remoto (RPC). Esse mecanismo pode ser usado em combinação com uma grande variedade de protocolos de rede tais como HTTP, SMTP, FTP, etc (CHESTER, 2001). O funcionamento do mecanismo de trocas de mensagens SOAP (request/response) é ilustrado através do exemplo de um Web Service proposto. A partir de um valor e de uma taxa de acréscimo, fornecidos pelo usuário, inseridos nos campos de um formulário eletrônico disponibilizado na Internet, que pode ser acessado de qualquer *browser*, é feito o seguinte cálculo: Resultado = valor + valor*(taxa de acréscimo / 100) sendo retornado ao usuário o valor do resultado .

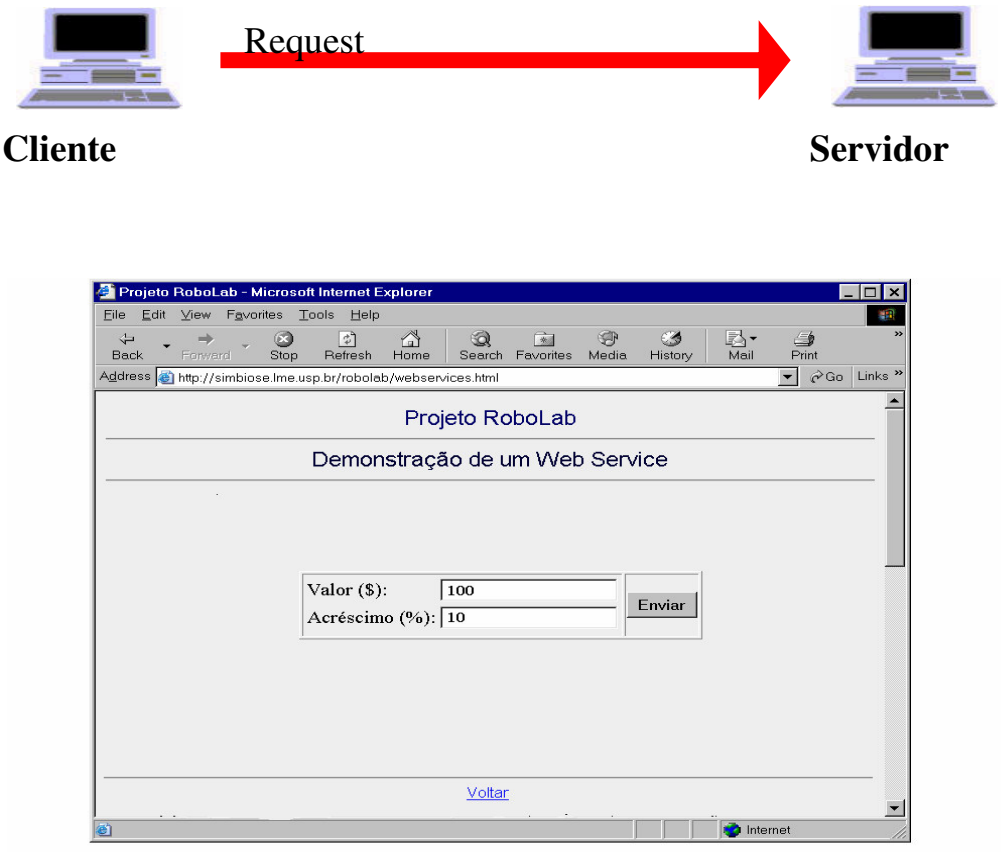

Figura 9. Formulário eletrônico de requisição do Web Service do exemplo proposto.

Figura 10. Resultado obtido do Web Service do exemplo proposto.

Na mensagem SOAP de requisição, enviada da aplicação cliente ao servidor, entre os parâmetros informados estão o nome do Web Service (*taxCalc* no exemplo) e o nome das variáveis (*sub* e *rate* referentes respectivamente ao valor e acréscimo), na qual o valor especificado pelo usuário das variáveis é delimitado por *tags* com o nome das variáveis.

```
POST /robolab/taxCalc.php HTTP/1.0
User-Agent: NuSOAP/0.6.3
Host: simbiose.lme.usp.br
Content-Type: text/xml; charset=ISO-8859-1
Content-Length: 573
SOAPAction: ""

<?xml version="1.0" encoding="ISO-8859-1"?>
<SOAP-ENV:Envelope SOAP-
ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:si="http://soapinterop.org/xsd">
      <SOAP-ENV:Body>
            <ns1:taxCalc xmlns:ns1="http://testuri.org">
                  <rate xsi:type="xsd:string">10</rate>
                  <sub xsi:type="xsd:string">100</sub>
            </ns1:taxCalc>
      </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Figura 11. Mensagem SOAP de requisição do *Web Service*.

Na mensagem SOAP de resposta, enviada do servidor ao cliente, como pode ser observado na fig. 12 a resposta do cálculo matemático do *Web Service taxCalc* é delimitado pela *tag* `<taxCalcResponse>` (no caso do exemplo o valor da resposta é 110,00), é interessante observar que o tipo da variável é especificado através do atributo `xsi:type` no caso do exemplo temos uma variável do tipo *float* (`"xsd:float"`).

O protocolo SOAP ainda permite que sejam enviadas mensagens de erro em determinadas situações (no caso do exemplo se o usuário digitar algum número negativo), estas mensagens são conhecidas como SOAP *fault* e é composta por quatro elementos, sendo que os dois primeiros são obrigatórios (CAMPBELL, 2002):

- *faultcode* – indica o tipo da falha podendo ser: Client, Server, MustUnderstand ou VersionMismatch.
- *faultstring* – indica ao usuário qual é a falha. Através de um texto feito pelo programador é possível inclusive informar ao usuário os procedimentos necessários para a correção do erro.
- *faultactor* – indica em qual Web Service ocorreu a falha. Somente é utilizada quando se está utilizando mais de um.
- *faultdetail* – elemento opcional para um maior detalhamento da falha.

```
HTTP/1.1 200 OK
Date: Wed, 05 Nov 2003 15:16:02 GMT
Server: NuSOAP Server v0.6.3
Accept-Ranges: bytes
X-Powered-By: PHP/4.2.2
Status: 200 OK
Connection: Close
Content-Length: 530
Content-Type: text/xml; charset=UTF-8
<?xml version="1.0" encoding="ISO-8859-1"?>

<SOAP-ENV:Envelope SOAP-ENV:
encodingStyle=http://schemas.xmlsoap.org/soap/encodi
ng/
xmlns:SOAP-
ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance
xmlns:SOAP-
ENC="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:si="http://soapinterop.org/xsd">
     <SOAP-ENV:Body>
          <taxCalcResponse>
               <noname xsi:type="xsd:float">110.00</n
          oname>
          </taxCalcResponse>
     </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Figura 12. Mensagem SOAP de resposta do web service.

5.4. Linguagem WSDL

A linguagem *Web Services Description Language* (WSDL) descreve os serviços disponibilizados para a rede através de uma semântica XML, que fornece a documentação necessária para se requisitar um sistema distribuído e o procedimento necessário para que esta comunicação se estabeleça. Enquanto o SOAP especifica a comunicação entre um cliente e um servidor, o WSDL descreve os serviços oferecidos. O WSDL tem uma função similar ao IDL do CORBA ou a implementação de uma interface remota em Java RMI (GUNZER, 2002).

O WSDL provê uma maneira de descrever Web Services com detalhes suficientes para serem acessados por pontos finais na rede. Os documentos WSDL consistem em cinco elementos principais que se combinam para descrever um Web Service. Dentre eles os três primeiros definem a mensagem e os dois últimos definem o protocolo e as informações de endereço. Estes elementos são:

- type: utilizado para definir a formatação da mensagem através dos tipos dos dados;
- messages: fornece detalhes das mensagens e parâmetros da mensagem, através da definição das informações de entrada e saída para uma determinada operação;
- portType: agrupa as mensagens nas operações executadas pelo Web Service, definindo as mensagens de entrada e saída;
- binding: descreve para cada mensagem definida no elemento portType o protocolo de transporte a ser utilizado;
- service: agrupa um ou mais elementos de definição que determinam o endereço específico de cada uma das operações do Web Service.

5.5. Web Service Discovery

O *Web Service Discovery*, também conhecido como *disco*, é processo de localização e informação das definições de um Web Service específico, que é uma etapa preliminar para utilizar um Web Service. É por meio desse processo de

descoberta que os usuários de um Web Service ficam sabendo que ele existe, quais são seus recursos e como interagir corretamente com ele.

Esse processo é viabilizado através da geração de um arquivo XML composto dos seguintes elementos:

- discovery: define o cabeçalho XML informando a localização da URL que define os elementos empregados neste arquivo XML;
- contractRef: informa a URL que disponibiliza o esquema WSDL, também pode informar uma URL que contenha de algum documento de auxílio ao usuário para a utilização do Web Service;
- discoveryRef: de utilização opcional, este elemento pode ser utilizado para apontar a localização de outros arquivos de *disco*;
- schemaRef: estabelece qual a especificação de XML que está sendo empregada;
- soap: com objetivo de agilizar a localização de um Web Service utilizando informações simples este elemento reproduz parte das informações contidas no WSDL.

5.6. Aplicabilidade da arquitetura Web Services à Instrumentação Inteligente

A possibilidade de integração de uma rede de sistemas embarcados dedicados à instrumentação a uma infra-estrutura de monitoramento via Web Services tem sido o objeto de estudo recente de vários grupos de pesquisa (JAMMES, 2005). Observa-se nestes trabalhos uma preocupação recorrente em estabelecer uma integração fundamentada em sistemas abertos e padronizados baseados nas tecnologias da Internet a fim de garantir a interoperabilidade. Alguns autores ressaltam ainda a problemática da integração "inteligente" dos dispositivos na rede, isto é, com dispositivos *plug-and-play* (JAMMES, 2005).

Dentro deste conceito, uma das primeiras iniciativas foi a arquitetura UPnP (Universal Plug and Play), formada por um consórcio de empresas em 1999, tendo como objetivo criar um padrão de comunicação comum entre dispositivos eletrônicos genéricos (computadores, televisores, automação predial, etc) via Internet. A arquitetura UPnP

utiliza várias tecnologias que também são utilizadas na arquitetura Web Services (IP, TCP, UDP, HTTP, SOAP e XML) mas que não são compatíveis entre si (UPnP FORUM, 2005). Atualmente a arquitetura UPnP tem sido mais utilizada em computadores e em dispositivos de redes de computadores.

No caso de uma rede de sensores com um esquema de comunicação ponto a ponto, é necessário que o sistema de monitoramento envie uma mensagem requisitando os dados endereçada ao sensor e que aguarde a resposta, como é o caso em uma arquitetura cliente/servidor. Este esquema de comunicação entre os sensores pode se tornar inviável devido à quantidade de mensagens que deverão circular na rede de sensores, principalmente no que se refere à transmissão de dados.

Os esquemas de comunicação multipontos comumente utilizados são os do tipo multicast e broadcast. No multicast a comunicação é 1 para N ou seja, permite enviar pacotes IP para um determinado grupo de usuários que previamente se cadastraram neste grupo, enquanto que no broadcast a comunicação é 1 para todos, sem a necessidade de um cadastro prévio. Normalmente estes esquemas de comunicação são implementados em protocolos de comunicação não orientados à conexão, como por exemplo, o protocolo UDP. A implementação de um esquema multicast tem uma complexidade extra que é a implementação de uma "inteligência" na rede para saber em quais portas existem usuários cadastrados em grupos, enquanto que o broadcast apresenta uma maior simplicidade de implementação. Por outro lado, esta simplicidade leva a uma ausência de seletividade, pois todos os elementos da rede recebem a informação e necessitam processá-la para verificar se é destinada a si.

Esta limitação do esquema broadcast pode ser resolvida através de uma arquitetura do tipo editora/assinante (*publisher/subscriber*) na qual é necessário que cada elemento faça uma assinatura prévia dos tipos de informação que deseja receber, num procedimento semelhante ao assinante de uma revista. Desta forma não é necessário que cada mensagem recebida seja analisada em cada elemento quanto ao conteúdo, para que se possa determinar se a mesma é útil ou não. A norma IEEE 1451.1 que define a interface de um transdutor inteligente (NCAP) com a rede prevê no seu modelo de comunicação a utilização tanto da arquitetura cliente/servidor quanto editora/assinante.

O modelo de comunicação do NCAP permite a utilização de ambas arquiteturas, pois imagina que dependendo da operação a ser executada uma é mais adequada que a outra, como por exemplo:

- Cliente / servidor: para a configuração do sistema de aquisição de dados associado a um transdutor ou solicitação de uma ação específica, isto é, tarefas que exigem uma maior confiabilidade;
- Editora / assinante: transmissão de dados dos sensores ou informação de identificação de transdutor conectado à rede.

Em um recente estudo da Sun Microsystems (HORAN, 2005) é feita uma classificação, sob outro aspecto, das tecnologias disponíveis ao desenvolvimento de redes de sensores inteligentes, no que se refere principalmente à capacidade de descrição das suas capacidades:

- Sintaxe: seriam as tecnologias que têm como ênfase em descrever a identidade do dispositivo (como por exemplo, se o dispositivo conectado na rede é um sensor de temperatura ou um sensor de pressão), bem como descrever o formato do dado retornado por este (como por exemplo, se o dado retornado é um *long* ou *boolean*);
- Semântica: enfatiza a descrição da funcionalidade do dispositivo em termos da aplicação real, através de um conjunto de protocolos de uma arquitetura que permite a descoberta dos serviços retornados pelo dispositivo (como por exemplo, no caso de um sensor de temperatura se a sua função seria medir a temperatura de uma caldeira ou de uma câmara frigorífica).

5.7. Conclusões do capítulo

O conceito de TEDS, da norma IEEE 1451, é um exemplo de uma tecnologia voltada à sintaxe, pois especifica com bastante detalhes a identificação de um sensor, com especial ênfase nas características elétricas deste sensor, enquanto a arquitetura Web Services, com o conceito de modelagem orientada a serviço, pode ser classificada como uma tecnologia focada na semântica, apresentando os serviços disponibilizados pelo dispositivo.

Os estudos citados anteriormente não estabelecem uma contraposição entre estes dois tipos de tecnologias, mas ressaltam a complementaridade entre as tecnologias e colocam como desafio a combinação delas para o sucesso na implementação de uma rede de sensores inteligentes.

6. INSTRUMENTAÇÃO INTELIGENTE VIA WEB SERVICES

Neste trabalho é proposto um primeiro passo para a integração de um sistema de instrumentação inteligente estabelecido na norma IEEE 1451.1 com a arquitetura Web Services. É proposta a associação de um protocolo de transmissão de dados associado ao conceito de TEDS a um web service que permita a configuração remota do sistema de aquisição de dados.

Através de uma interface de supervisão remota, acessível via Internet a partir de qualquer navegador, o usuário é capaz de em um primeiro momento identificar quais são os sensores inteligentes que estão conectados em uma rede, através de uma identificação única associada às suas características (tipo, formatação dos dados, etc.). Nessa mesma interface de apresentação, o usuário seleciona quais são os sensores de monitoramento que lhe interessam. Após a seleção é disponibilizada a interface de monitoramento que permite a visualização dos dados através de um gráfico, bem como permite ao usuário salvar em um arquivo de texto os dados coletados. Na mesma interface de monitoramento é disponibilizado o web service que permite alterar o intervalo de coleta de dados.

O ambiente de desenvolvimento deste trabalho é composto por um conjunto de computadores em uma rede local (Intranet) simulando uma rede de sensores inteligentes, cada um associado a um NCAP. Nesta mesma rede está conectado o servidor responsável pela disponibilização da interface de supervisão remota. Na transmissão dos dados para o servidor principal foi utilizada a porta 8080, enquanto que para o controle via Web Service e publicação dos dados na Internet foi utilizada a porta 80, conforme pode ser observado na figura 13.

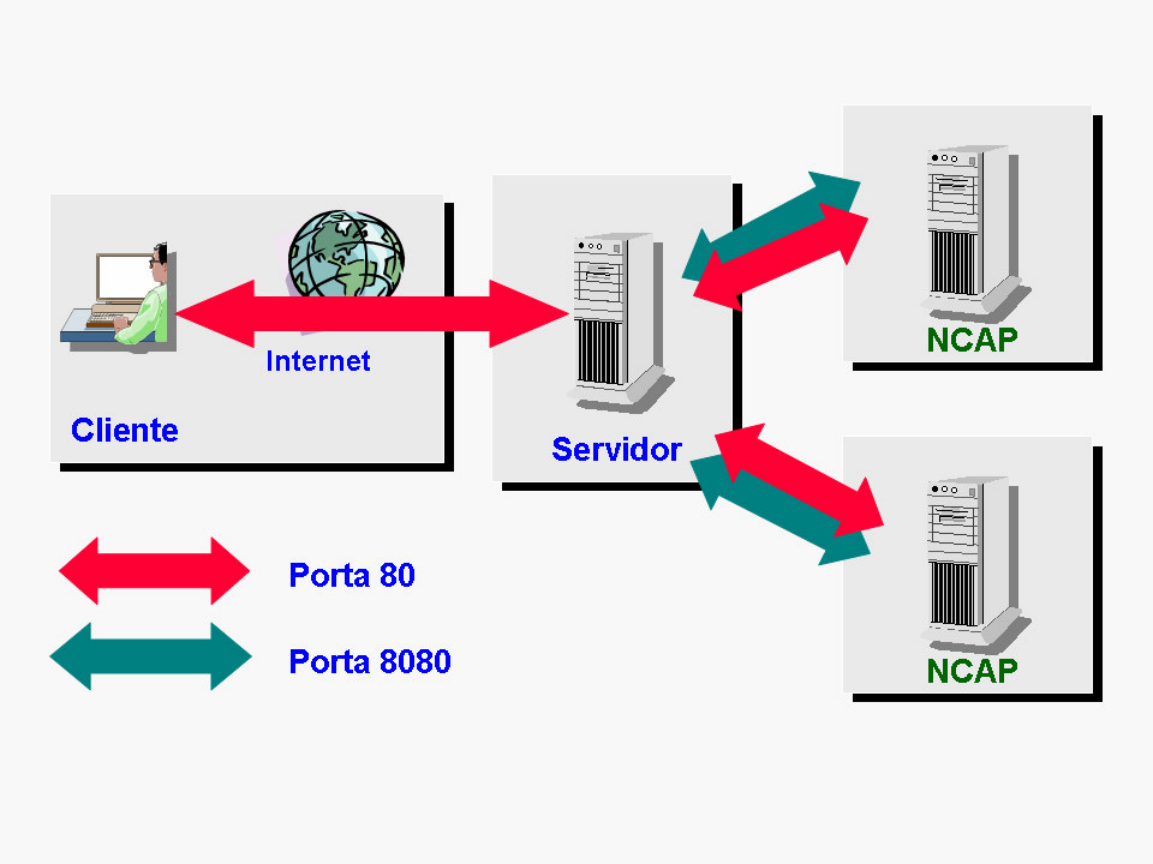

Figura 13 Diagrama esquemático do ambiente de desenvolvimento.

6.1. Materiais e métodos

6.1.1. JDDAC

A simulação da rede de sensores inteligentes e a comunicação dos dados seguindo as diretrizes da norma IEEE 1451 foram desenvolvidas utilizando o Java Distributed Data Acquisition and Control (JDDAC). Esta ferramenta compreende um conjunto de API's *open source* que permite a conexão de aplicações em Java com sistemas de instrumentação que seguem as normas IEEE 1451, em específico as normas 1451.1 e 1451.2. Este projeto é desenvolvido por um grupo de pesquisadores dos Departamentos de Ciência da Computação, Engenharia Elétrica e Estudos Ambientais da San Francisco State University, com apoio dos centros de pesquisa da Agilent e Sun Microsystems, sendo que a primeira versão foi lançada em Março de 2004. Uma das motivações do

grupo foi contribuir com uma lacuna existente na plataforma Java no que se refere à interação com o mundo real, seguindo algum tipo de padronização que permitisse a perfeita interoperabilidade entre diversos sistemas de instrumentação. Atualmente o grupo de desenvolvimento publicou a última versão do JDDAC (versão 0.5) e trabalha no desenvolvimento de um sistema de monitoramento ambiental, das águas da Baía de San Francisco, empregando uma rede de sensores sem fio associada ao JDDAC (JDDAC, 2005).

O JDDAC define métodos para descrever o dispositivo final do sistema de instrumentação, indicando como eles e os dados gerados por eles podem ser tratados por uma aplicação Java. Estas abstrações reduzem o tempo de desenvolvimento, pois evitam a necessidade de se desenvolver um sistema a partir de API's de baixo nível, como por exemplo, as que tratam da comunicação serial.

O emprego de metadados descrevendo as características da medição (tipo de sensor, unidade, formato do dado, etc) efetuada pelos sensores, associando-os aos próprios dados da medição, é realizado pelo JDDAC, utilizando-se o XML, mas com o cuidado de que estes metadados não são gerados pelo próprio código-fonte, o que inviabilizaria a flexibilidade do sistema.

O JDDAC tem uma arquitetura centrada nos dados na qual há um modelo computacional baseado no fluxo de dados, atrelado à disponibilidade de dados e não há uma seqüência de instruções, o que o torna um modelo mais realista e de fácil reutilização. É interessante observar que o LabVIEW tem uma arquitetura similar o que explica a sua popularidade pois a sua estrutura de programação acaba se tornando bastante intuitiva para um desenvolvedor que conheça a forma como o processo de medição é realizado no mundo real.

As API's do JDDAC são divididas nas seguintes categorias:

- *Java Transducer API* : estabelece a interface dos TEDS com o JDDAC de maneira a identificar o transdutor e a forma de medição;
- *Java Measurement Calculus API*: formata os dados provenientes das medições, bem como o resultado de cálculos matemáticos com estas medições, para uma representação comum;

- *Java Measurement Dataflow API*: estabelece um modelo de fluxo de dados pelo qual os dados são processados e convertidos;
- *Java Precision Clock Synchronization API*: especifica a interface de gerenciamento e utilização do clock de sincronismo dos diversos pontos de medição de um sistema de instrumentação em rede.

O emprego de metadados através do XML no JDDAC torna a sua associação com o Web Services, que emprega o protocolo SOAP / XML, uma opção viável em termos de assegurar uma arquitetura com modularidade e interoperabilidade.

6.1.2. PHP

Para o desenvolvimento da interface de supervisão remota foi utilizada a linguagem PHP. O PHP é uma linguagem de scripts, de uso geral, amplamente utilizada no desenvolvimento de aplicações para a Internet. Os scripts podem ser inseridos em um código HTML, com uma sintaxe semelhante ao C, Perl e Java.

A linguagem PHP foi concebida em 1994 por Rasmus Lerdorf, sendo que as primeiras versões foram disponibilizadas em 1995. A linguagem ficou conhecida inicialmente como *Personal Home Page Tools* (ferramentas para página pessoal). Era composta por um sistema bastante simples que interpretava algumas macros e alguns utilitários que eram executados de forma oculta nas home-pages: como um livro de visitas, um contador e outros aplicativos. Em meados de 1995 o interpretador foi reescrito e recebeu o nome de PHP/FI, no qual foi acrescentado um outro pacote escrito por Rasmus que interpretava dados de formulários HTML (*Form Interpreter*) e adicionava um suporte à base de dados mySQL. A popularização desta versão foi imediata. Estima-se que em 1996 ela estava sendo usada em cerca de 15.000 sites da Internet, e em meados de 1997 esse número subiu para mais de 50.000. Nessa época, uma equipe de desenvolvimento assumiu o projeto PHP, o que levou o interpretador a ser reescrito. Este novo interpretador foi a base para a versão 3. Com o PHP 4, lançado em 2000, alguns novos recursos foram acrescentados como suporte a sessões, bastante útil para identificar o cliente que solicitou determinada informação, bem como mudanças referentes à sintaxe e novos recursos de programação, esta nova versão foi lançada com um otimizador

chamado Zend, que permite a execução muito mais rápida de scripts PHP. Em 2004 é lançado o PHP 5, com uma série de alterações substanciais, baseado totalmente em uma arquitetura orientada a objeto. As alterações dos conceitos de programação nesta última versão foram tão grandes que a versão 4 ainda é mantida em atividade com atualizações desenvolvidas em paralelo com as atualizações da versão 5.

Uma página na Internet desenvolvida em HTML puro, que é usado como uma linguagem descritiva do formato de apresentação dos dados contidos no servidor Web, é capaz, por exemplo, de introduzir links, selecionar o tamanho das fontes ou inserir imagens, tudo isto de uma maneira pré-estabelecida. Com ele podemos criar somente páginas estáticas. Ele não é capaz de realizar um cálculo matemático ou criar uma página nova a partir de uma base de dados.

O PHP é uma linguagem que permite criar páginas dinâmicas, isto é, páginas que podem alterar a sua formatação de apresentação de acordo com a mudança de determinadas variáveis, como exemplo dados inseridos pelo usuário através de um formulário, recebidos por um socket, etc. Este processo de automatização da geração de uma página segue um "script" definido pelo programador.

A diferença do PHP com relação a linguagens semelhantes como o Javascript, por exemplo, é que o código PHP é executado no servidor, sendo enviado para o cliente apenas o código HTML. Dessa maneira, é possível interagir com bancos de dados e aplicações existentes no servidor, com a vantagem de não expor o código fonte para o cliente. Isso pode ser útil quando o programa está lidando com senhas ou qualquer tipo de informação confidencial.

O que diferencia o PHP de um script CGI escrito em C ou Perl é que o código PHP fica embutido no próprio HTML, enquanto no outro caso é necessário que o script CGI gere todo o código HTML, ou leia de um outro arquivo. Basicamente, qualquer coisa que pode ser feita por algum programa CGI pode ser feita também com PHP, como coletar dados de um formulário, gerar páginas dinamicamente ou enviar e receber *cookies*.

A utilização de páginas dinâmicas em sistemas de instrumentação remota via Internet é pré-requisito, pois o usuário deve ser capaz de interagir remotamente com o dispositivo eletrônico de instrumentação, como por exemplo para ajustar os *set-up's* da

coleta de dados. Além do mais o usuário espera visualizar aqueles dados coletados de alguma maneira, como por exemplo, através de um gráfico, o que não é uma operação estática.

A literatura apresenta vários exemplos de sistemas de instrumentação remota via Internet desenvolvidos com CGI (SUZUKI, 1996) (SHAHEEN, 1998), a sua utilização nestes sistemas apresenta algumas desvantagens como a dependência de plataforma e a perda de eficiência devido à necessidade de se carregar e processar o script para cada requisição de um cliente (QIJUN, 2003).

No trabalho aqui apresentado optou-se pelo PHP por ser uma ferramenta consolidada no desenvolvimento de páginas dinâmicas, pela independência de plataforma, facilidade de manutenção e eficiência.

O PHP também tem como uma das suas características mais importantes o suporte a um grande número de bancos de dados, como Interbase, mySQL, Oracle, PostgreSQL entre outros. Com o PHP ainda é possível abrir *sockets,* interagir com outros protocolos e inserir no seu script linhas de código desenvolvidas em outra linguagem. Por ser um software livre, constantemente novas funcionalidades são acrescidas ao PHP, um exemplo é NuSOAP que permite o desenvolvimento de Web Services com o PHP. O NuSOAP foi utilizado para o desenvolvimento do Web Service para a configuração remota do sistema de aquisição de dados.

6.1.3. NuSOAP

O NuSOAP é um conjunto de classes do PHP que permite aos usuários enviar e receber mensagens SOAP sob o protocolo HTTP, ele não é uma extensão do PHP mas é escrito em puro PHP . Desenvolvido por Dietrich Ayala e distribuído pela NuSphere Corporation, o NuSOAP é um software livre licenciado pelo GNU LGPL.

No NuSOAP ao mesmo tempo em que é implementada uma funcionalidade ela é associada a um servidor SOAP, como no exemplo apresentado em que ao mesmo tempo em que é criada uma função de geração de números aleatórios ela é associada ao web service intitulado *geraNumAleatorio*.

```
<?php
        // incluir classe NuSOAP
        require('nusoap.php');
        // instância do servidor SOAP
        $s = new soap_server;
        $s->register('geraNumAleatorio');
        function geraNumAleatorio($min, $max) {
                srand((double)microtime()*1000000);
                $numAleatorio = rand($min, $max);
                return $numAleatorio;
        }
        $s->service($GLOBALS["HTTP_RAW_POST_DATA"]);
?>
```

Figura 14. Servidor SOAP contendo a função de geração de números aleatórios.

Quanto ao desenvolvimento do cliente SOAP existem duas camadas distintas: a de interface com o usuário, que segue as diretrizes de uma página dinâmica qualquer desenvolvida com PHP e HTML, e a camada composta pelas classes responsáveis pela ligação ao web service e pelo tratamento dos dados.

```
<?php
        // incluir classe NuSOAP
        require('nusoap.php');
        $min = $_REQUEST["minimo"];
        $max = $_REQUEST["maximo"];
        $localizacaoServidor = 'http://localhost:8080/server.php';
        $parameters = array('min'=>$min,'max'=>$max);
        // instância do cliente SOAP
        $soapclient =& new soapclient($localizacaoServidor);
        $resultado=$soapclient->call('geraNumAleatorio',$parameters);
?>
```

Figura 15. Cliente SOAP contendo a função de geração de números aleatórios.

O NuSOAP estabelece um Web Service através de uma classe chamada *soapclient*, instanciada a partir da localização do script contendo o servidor SOAP, composta por um conjunto de métodos responsáveis pelo gerenciamento das mensagens SOAP. Um destes métodos é o *call*( ), nele é informado o Web Service que será requisitado, bem como os parâmetros para a sua execução, no caso do exemplo a especificação dos valores máximos e mínimos dos números aleatórios. O método então retornará a resposta do servidor.

Em uma análise comparativa entre o NuSOAP para PHP, SOAP-Lite para Perl, WASP Server para Java e .NET com recurso a linguagem C#, os seguintes resultados foram observados (AYALA,2003):

- no NuSOAP a criação do servidor SOAP é bem simples e é feita ao mesmo momento no qual se definem as funcionalidades desse servidor, tal como no .NET;
- ao mesmo tempo em que se cria um cliente SOAP, é requisitado o Web Service através do método *call* ( ), com o envio dos parâmetros. Essa metodologia é semelhante a metodologia empregada no SOAP-Lite.

Como se pode observar a simplicidade de desenvolvimento de um Web Service com o NuSOAP é observada tanto no desenvolvimento da aplicação cliente quanto da aplicação servidor.

6.2. Desenvolvimento

6.2.1. Especificação do protocolo de comunicação

O projeto JDDAC, por ser recente, ainda está em fase de desenvolvimento por parte da sua comunidade de desenvolvedores. O JDDAC especifica com clareza a forma como pode ser implementada uma aplicação cliente, que é responsável pela identificação do sensor inteligente e pela aquisição e transmissão dos dados provenientes deste sensor. Em outras palavras, em termos da norma IEEE 1451, ela exerce o papel do NCAP. Por outro lado, a aplicação servidor, responsável pela coleta e apresentação desses dados, bem como pelo gerenciamento da comunicação dos sensores, até o presente momento

não foi definida pela comunidade dos desenvolvedores do JDDAC. Foi criada uma aplicação servidor para o JDDAC neste mestrado, tendo como referência a arquitetura que inicialmente foi proposta e o modo de operação do protocolo de comunicação da aplicação cliente do JDDAC.

O protocolo de comunicação de dados do NCAP no JDDAC emprega o XML, dentro de uma estrutura hierarquicamente organizada, mas com uma particularidade que o diferencia de outras aplicações que também empregam XML. Consiste no emprego dos metadados (tipo de sensor, unidade, formato do dado, etc) como conteúdo das tags, juntamente com os próprios dados da medição, ao invés do seu emprego na própria tag, como normalmente é feito, justamente para que o protocolo de comunicação seja genérico, isto é, seja independente do tipo de sensor empregado. Neste protocolo de comunicação são empregados somente duas tags, conforme indicado na tabela.

Tag	***Descrição***
argArray	Define um elemento-raiz que referencia os elementos subseqüentes dentro da hierarquia.
arg	Define o elemento que contém o metadado ou o dado.

Tabela 2. Conjunto de *tags* empregados no protocolo de comunicação.

Um conjunto de atributos, cada qual consistindo de um nome e um valor, pré-definidos são associados a cada tag.

Atributos	***Descrição***
l ***(lenght)***	Indica o número de elementos-filhos associados ao elemento do atributo.
n ***(name)***	Atributo obrigatório que especifica o conteúdo da tag.
t ***(type)***	Atributo opcional que complementa a informação do atributo *name*.

Tabela 3. Conjunto de atributos empregados no protocolo de comunicação.

O código XML de comunicação na íntegra pode ser observado na figura 16. Uma das vantagens do XML é a facilidade de interpretação humana dos conteúdos, por ter quase que o formato de um texto em uma linguagem humana.

Os elementos do tipo *measurement* têm 8 elementos-filhos tendo como conteúdo as informações provenientes do sensor (metadados e dados), sendo que o elemento do tipo *location* estabelece a possibilidade de associação destas informações às coordenadas geográficas do sensor fornecidas por um dispositivo GPS embutido, mas este recurso não foi empregado.

A simulação dos sensores foi executada através de um conjunto de classes disponibilizadas pelo JDDAC. Dois sensores foram criados: um responsável por medir a quantidade de memória computacional utilizada pelo Java Virtual Machine que interpreta o *bytecode* da aplicação-cliente do JDDAC e o outro, responsável por medir a memória que ainda resta, ambas as grandezas medidas em bytes, em função do tempo.

```
<?xml version="1.0" ?>
- <argArray l="1" t="vector" n="http://192.168.210.106:8080">
  - <argArray l="2">
    - <argArray l="2" t="record" n="Value">
      - <argArray l="8" t="measurement" n="totalMemory">
        - <argArray l="5" t="location" n="location">
            <arg n="fix">0</arg>
            <arg n="datum">0</arg>
            <arg n="log">0.0</arg>
            <arg n="alt">0.0</arg>
            <arg n="lat">0.0</arg>
          </argArray>
          <arg n="units">Bytes</arg>
          <arg n="dataType">Integer64</arg>
          <arg n="name">totalMemory</arg>
          <arg n="description">JVM Total Memory</arg>
          <arg t="long" n="value">2031616</arg>
          <arg t="timeRepresentation"
            n="timestamp">1108498704.635000000</arg>
          <arg n="id">jddac://Quetzal/totalMemory</arg>
        </argArray>
      - <argArray l="8" t="measurement" n="freeMemory">
        - <argArray l="5" t="location" n="location">
            <arg n="fix">0</arg>
            <arg n="datum">0</arg>
            <arg n="log">0.0</arg>
            <arg n="alt">0.0</arg>
            <arg n="lat">0.0</arg>
          </argArray>
          <arg n="units">Bytes</arg>
          <arg n="dataType">Integer64</arg>
          <arg n="name">freeMemory</arg>
          <arg n="description">JVM Free Memory</arg>
          <arg t="long" n="value">1798792</arg>
          <arg t="timeRepresentation"
            n="timestamp">1108498704.635000000</arg>
          <arg n="id">jddac://Quetzal/freeMemory</arg>
        </argArray>
      </argArray>
      <arg n="Topic">end</arg>
    </argArray>
  </argArray>
```

Figura 16. Código XML de comunicação do NCAP.

O elemento *id* tem como conteúdo a identificação única que especifica a localização do NCAP. No caso deste experimento foi adotada como identificação o nome do computador (pelo qual ele está registrado na intranet) no qual está hospedado o NCAP. Além desta identificação está presente a identificação de qual grandeza está sendo medida pelo sensor.

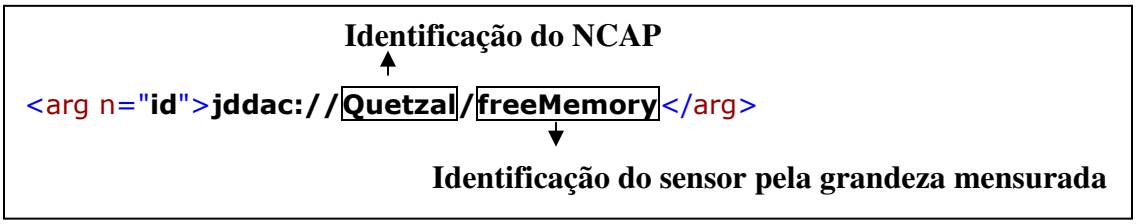

Figura 17. Parâmetro de identificação do código XML de comunicação do NCAP.

O XML de comunicação é enviado pelo NCAP ao servidor, identificado pelo seu IP interno e porta de comunicação registrados previamente no NCAP, através de um socket via TCP/IP. O servidor entra no modo de *listenig* (de escuta) na porta 8080, porta escolhida para receber a comunicação do NCAP, no momento em que é inicia a sessão de supervisão. A partir daí todos os NCAP´s enviam uma comunicação no mesmo formato apresentado anteriormente. Este processo permite ao servidor identificar quais são os NCAP´s que estão operando naquele momento, bem como os sensores associados a cada NCAP.

Cada NCAP fica no aguardo da solicitação por parte do servidor para o envio de mais um pacote de dados. Esta solicitação somente será feita pelo servidor caso o usuário deseje monitorar o NCAP em questão. A solicitação é executada pelo envio ao NCAP de uma mensagem, também no formato XML composta por uma única tag, sem nenhum conteúdo.

Para o gerenciamento do socket de comunicação foi empregado os próprios recursos disponibilizados pelo Linux, através da função *Netcat*.

```
<?xml version="1.0" ?>
 <argArray l="0" t="vector" n="jddac://Quetzal" />
```

Figura 18. Código XML de solicitação de envio da comunicação.

Esta mensagem segue um formato especificado pelo JDDAC e ela é recebida por todos os NCAP's, mas somente o NCAP que tiver a mesma identificação do que a registrada no atributo *name* (n) desta mensagem responderá com um novo pacote de dados.

6.2.2. Interface de supervisão

Foi desenvolvida uma interface de supervisão, ela é publicada na Internet pelo mesmo servidor responsável pela coleta dos dados enviados pelos NCAP's conectados à Intranet, com a diferença de que a publicação da interface, bem como toda a interação com o usuário, é estabelecida através da porta 80. A utilização desta porta evita possíveis problemas no que se refere a firewalls, e através desta interface (figura 19) são apresentados ao usuário todos os NCAP's identificados na rede interna que estão operando naquele momento, bem como os respectivos sensores associados a cada NCAP.

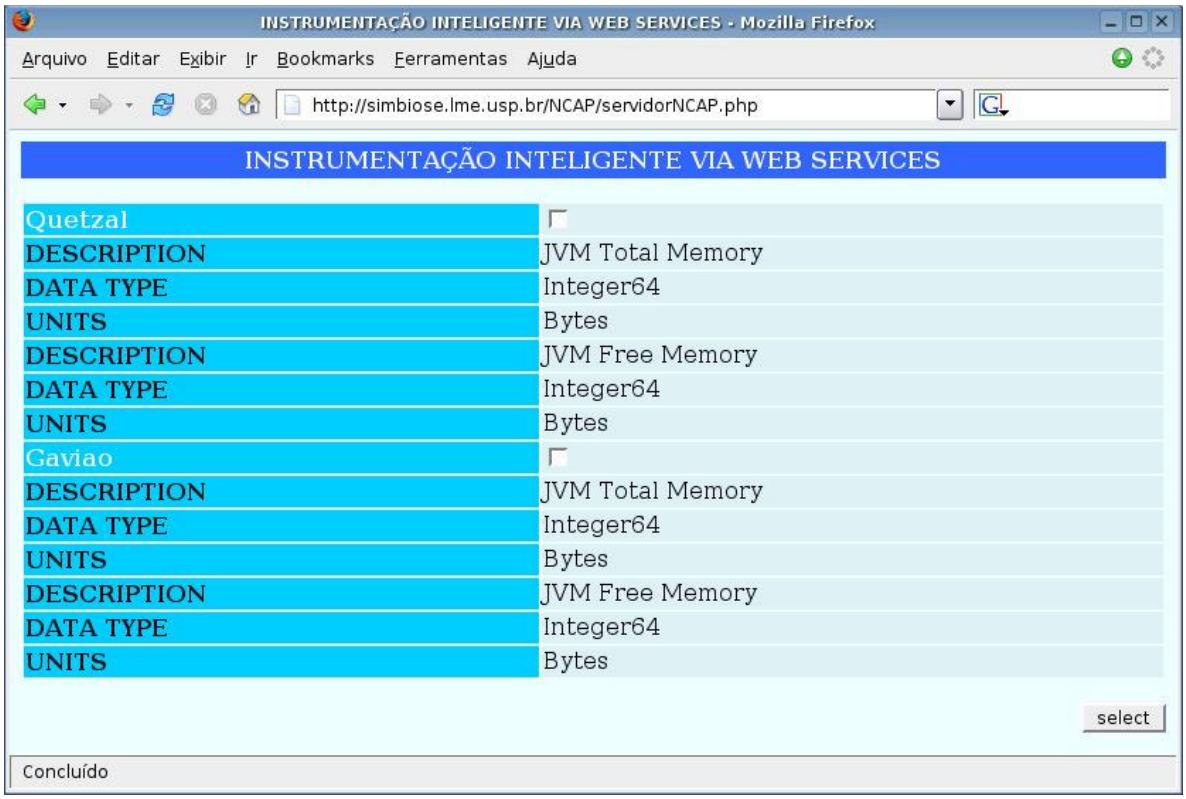

Figura 19. Interface de supervisão com a identificação de cada NCAP com os respectivos sensores associados.

Abaixo da identificação de cada NCAP são apresentados alguns campos com informações sobre cada sensor:

- **Description**: identifica a grandeza que está sendo medida pelo sensor;
- **Data type**: especifica o tipo do dado retornado pelo NCAP em termo de variável computacional (float, integer, boolean, etc);
- **Units:** informa a unidade de medida empregada pelo sensor

Através do *checkbox* localizado ao lado da identificação de cada NCAP o usuário pode selecionar quais são os NCAP's que são do seu interesse o monitoramento. Após a identificação dos NCAP's selecionados se estabelece o processo de comunicação contínua com eles através do envio das mensagens de solicitação de dados, bem como de

recepção dos dados. Após esta seleção é disponibilizada ao usuário a interface que permite o acesso à visualização e registro dos dados.

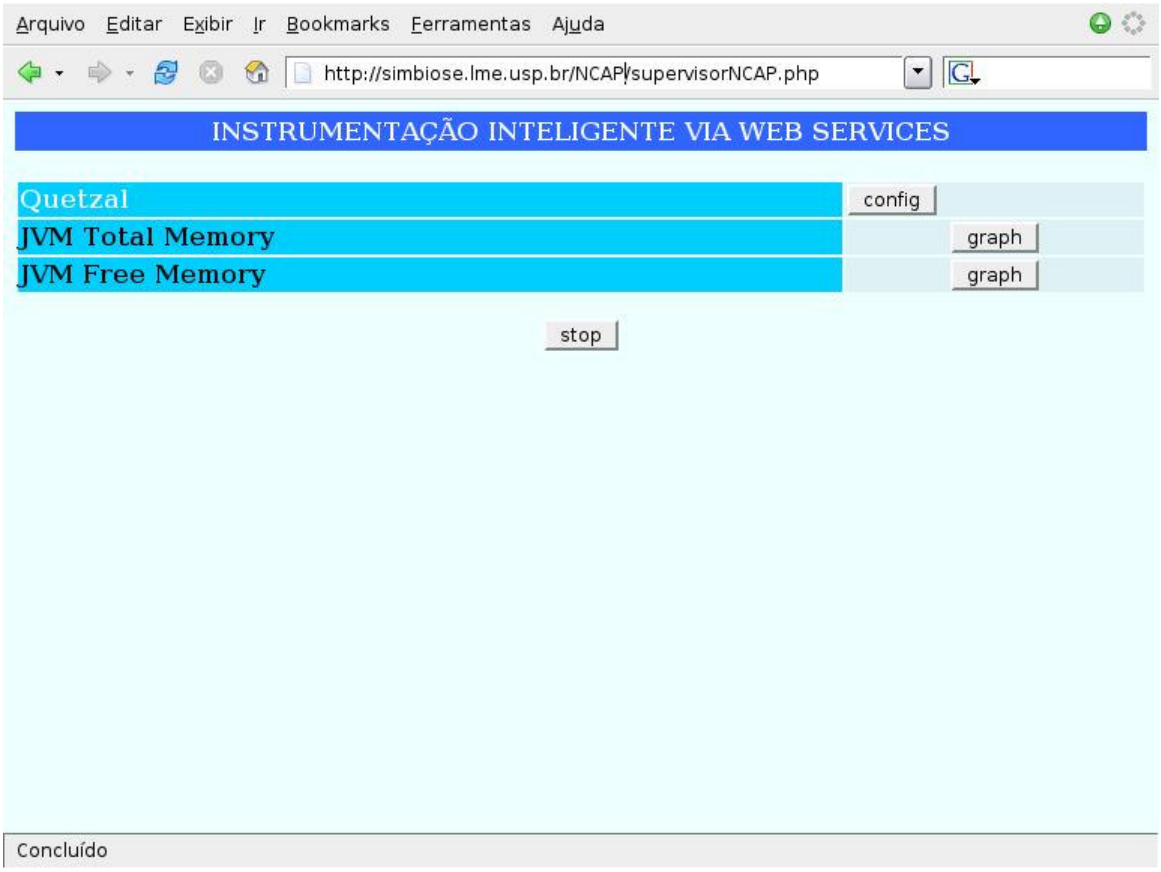

Figura 20. Interface de acesso à visualização e registro dos dados.

6.2.3. Publicação e registro dos dados.

Na interface de publicação e registro dos dados é apresentada uma listagem com todos os NCAP's que foram selecionados na interface anterior (figura 20), bem como a identificação de cada sensor associado a cada NCAP. Para cada sensor é disponibilizado um recurso de visualização dos dados que estão sendo coletados, através da função *graph*, pela qual é possível o monitoramento dos dados através de um gráfico (figura 21).

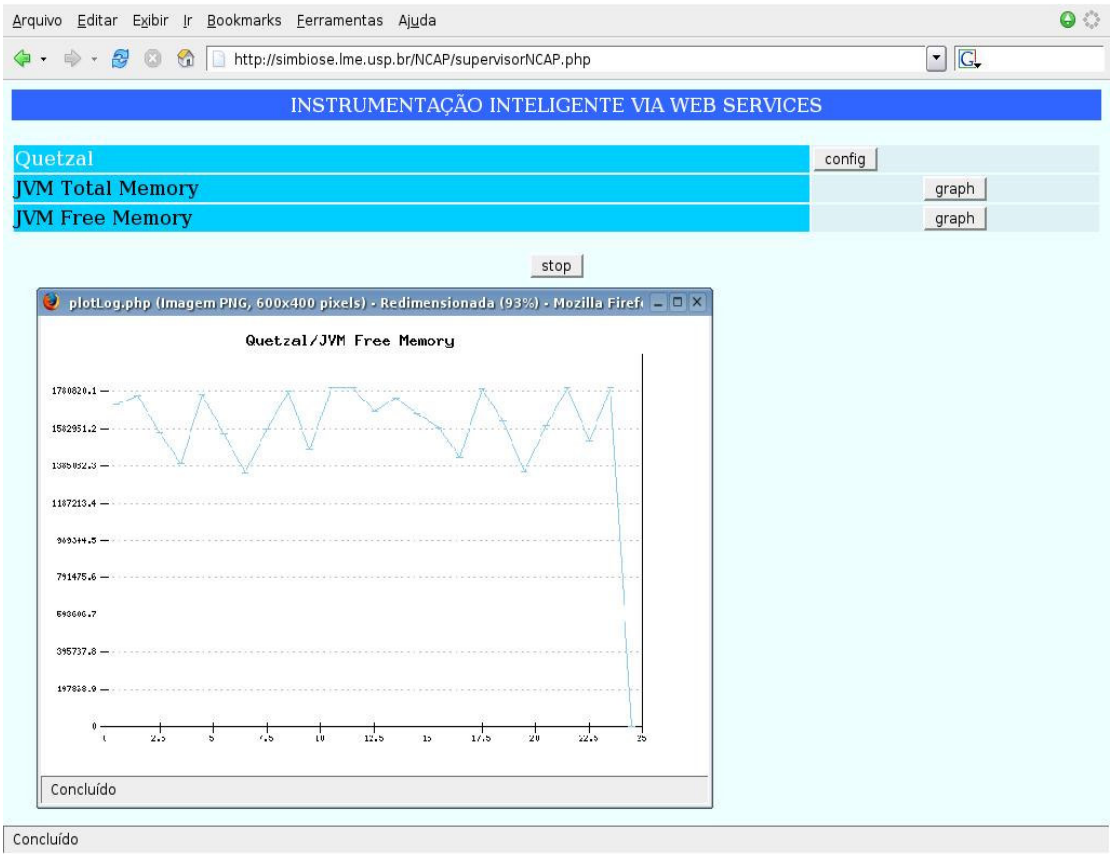

Figura 21. Interface de supervisão com a função *graph* acionada

A exposição dos gráficos permite uma apresentação que, em termos de interface homem-máquina, facilita o monitoramento de uma determinada grandeza, mas independente de que o monitoramento do sensor esteja sendo feito pelo usuário, através do gráfico, todos os dados recebidos são registrados pelo servidor.

Quando uma sessão de monitoramento é finalizada, pela função *stop* os dados registrados são formatados e apresentados através de um arquivo com extensão *txt,* que pode ser salvo no computador do usuário (figura 22).

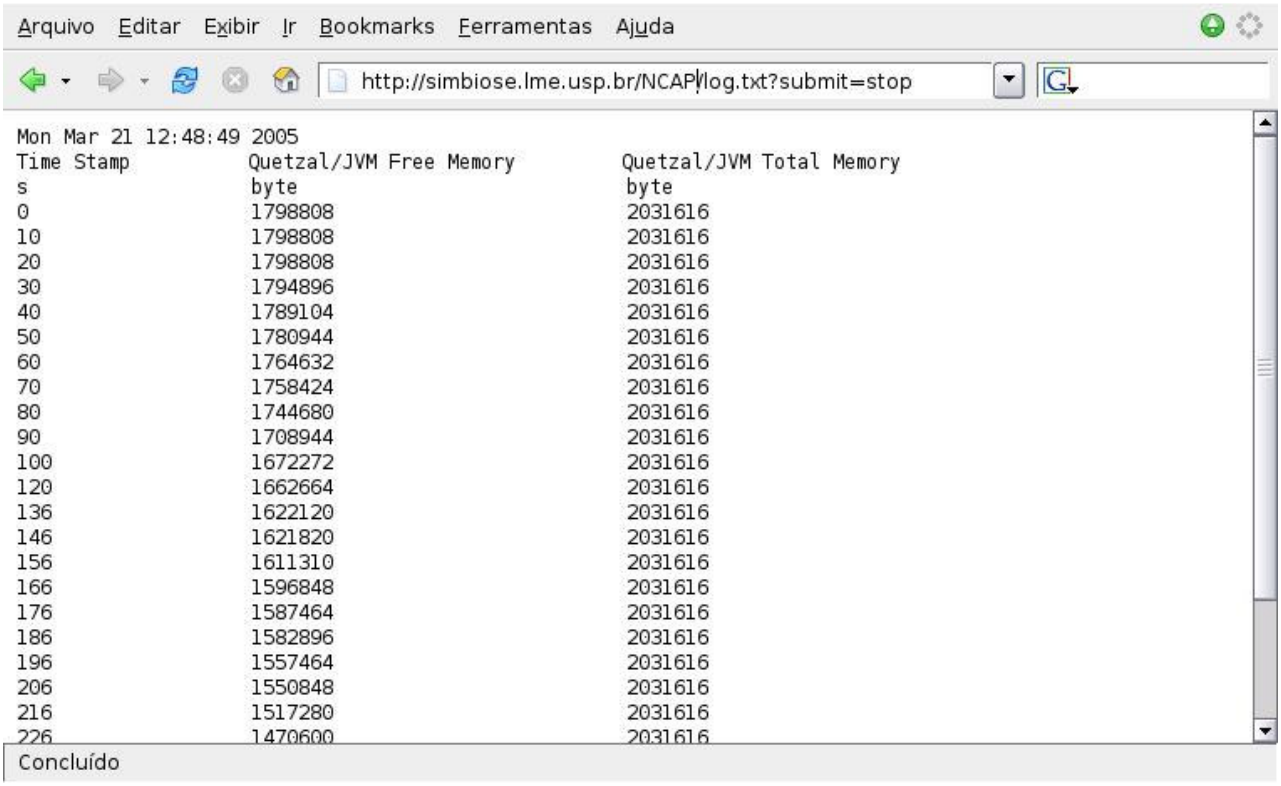

Figura 22. Visualização dos dados registrados e formatados

6.2.4. Web Service de configuração remota

Na mesma interface de publicação e registro dos dados, é disponibilizada, para cada NCAP, a função *config* que habilita o web service *configService* . Através de uma interface, o usuário pode selecionar, por meio de uma caixa de seleção, o intervalo de aquisição de dados (figura 23).

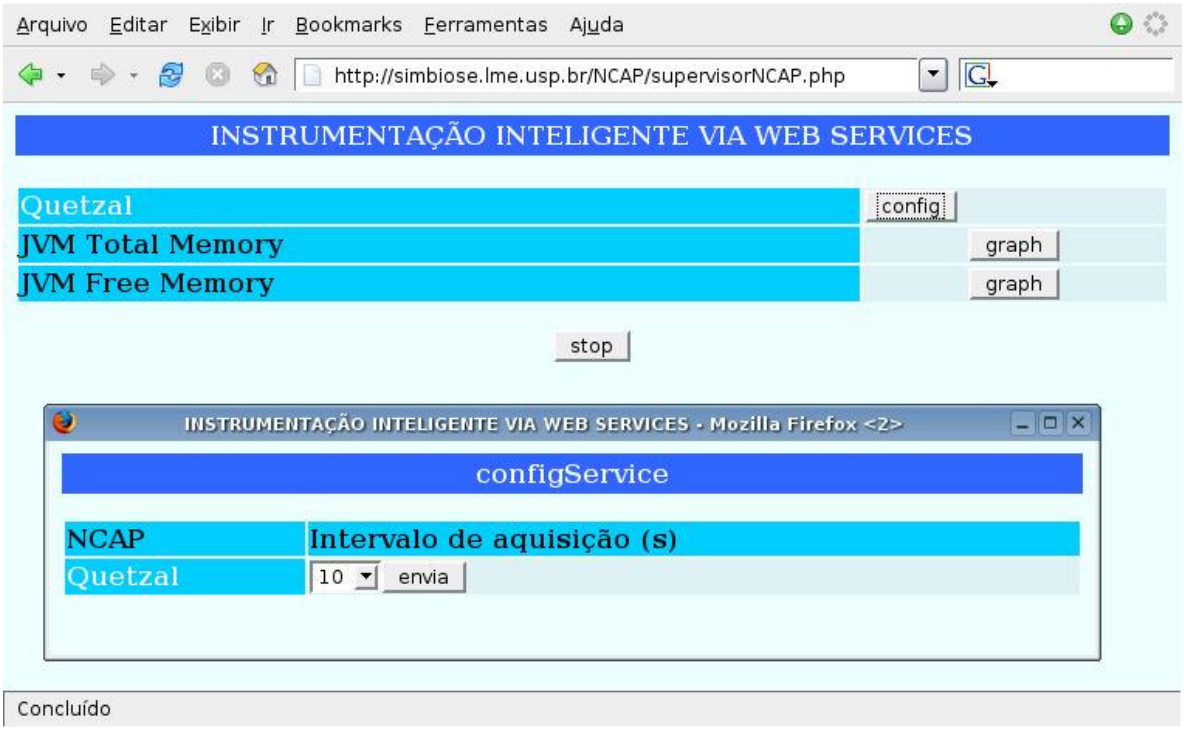

Figura 23. Interface de acesso ao web service ***configService.***

Após o envio do intervalo selecionado, uma solicitação SOAP (figura 24) é enviada ao servidor contendo o intervalo selecionado, no corpo da mensagem, tendo como atributo associado a identificação do NCAP que é o alvo da atuação. O servidor, através deste atributo, identifica o IP interno do computador que contém o NCAP, redirecionando a solicitação a este computador, que contém uma função (desenvolvida em PHP) que altera o arquivo de configuração do NCAP (no formato XML) para o intervalo desejado.

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
- <SOAP-ENV:Envelope SOAP-
    ENV:encodingStyle="http://schemas.xmlsoap.org/soap/enco
    ding/" xmlns:SOAP-
    ENV="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:SOAP-
    ENC="http://schemas.xmlsoap.org/soap/encoding/"
    xmlns:si="http://soapinterop.org/xsd">
  - <SOAP-ENV:Body>
      <ns8126:configTempo
        xmlns:ns8126="Quetzal">30</ns8126:configTempo>
    </SOAP-ENV:Body>
  </SOAP-ENV:Envelope>
```

Figura 24. Solicitação SOAP enviada ao servidor contendo o intervalo de aquisição selecionado.

Como forma de confirmação, é retornada ao usuário, na própria resposta à solicitação SOAP, uma mensagem SOAP (figura 25) contendo a leitura do parâmetro de controle do intervalo, já alterado, o que garante o sucesso da operação.

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
- <SOAP-ENV:Envelope SOAP-
    ENV:encodingStyle="http://schemas.xmlsoap.org/soap/enco
    ding/" xmlns:SOAP-
    ENV="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:SOAP-
    ENC="http://schemas.xmlsoap.org/soap/encoding/"
    xmlns:si="http://soapinterop.org/xsd">
  - <SOAP-ENV:Body>
    - <ns1:configTempoResponse xmlns:ns1="Quetzal">
  <return xsi:type="xsd:string"> &lt;arg t=&quot;long&quot;
n=&quot;Reporter.connectInterval&quot;&gt;30&lt;/arg&gt;</return>
      </ns1:configTempoResponse>
    </SOAP-ENV:Body>
  </SOAP-ENV:Envelope>
```

Figura 25. Mensagem SOAP de confirmação.

6.3. Discussões e resultados obtidos

Através dos resultados obtidos, conseguiu-se elaborar uma interface de supervisão remota via Internet de um conjunto de NCAP's, cada um por sua vez associado a um conjunto de sensores simulados. Nesta mesma interface foi disponibilizado um Web Service de configuração do NCAP. Dos resultados obtidos podem se destacar:

- foi criada uma aplicação-servidor para o JDDAC neste mestrado, tendo como referência a arquitetura que inicialmente foi proposta e o modo de operação do protocolo de comunicação da aplicação cliente do JDDAC, única aplicação disponibilizada pela comunidade de desenvolvedores do JDDAC;
- nesta aplicação foi inserido um Web Service que disponibiliza um serviço de configuração remota dos NCAP's, que pode ser solicitado através de uma mensagem seguindo o protocolo SOAP.

Os resultados obtidos respeitaram a necessidade de uma arquitetura com modularidade e interoperabilidade.

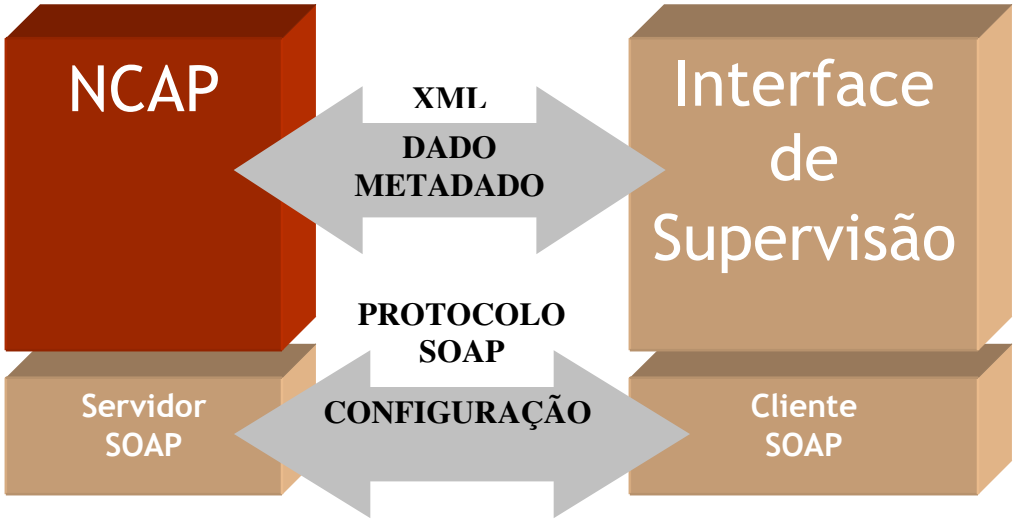

Figura 26. Diagrama conceitual dos resultados obtidos.

A aplicação-servidor para o JDDAC, criada neste mestrado, foi desenvolvida com o PHP. O JDDAC é desenvolvido com a linguagem Java a sua convivência com uma plataforma desenvolvida em PHP foi pacífica, graças a utilização de protocolos abertos que permitem uma perfeita interoperabilidade, como a tecnologia Web Services e XML.

A interface de interação com a aplicação NCAP / Servidor SOAP estabelecida através dos protocolos SOAP / XML permite uma independência de plataforma da Interface de Supervisão / Cliente SOAP, que pode ser facilmente migrado para outra plataforma, como por exemplo, Java ou .NET.

Para a extração dos dados, recebidos no formato XML, foi utilizada a biblioteca EXPAT (Parser XML) do PHP. O EXPAT é uma biblioteca *open-source* desenvolvida em C, e também é utilizada em outras linguagens de programação, como Perl, Python, entre outras.

Esta opção é vantajosa enquanto permite que :

- dados de fontes diferentes sejam consolidados em um único documento;
- o navegador não precisa suportar o XML;
- uma flexibilidade de desenvolvimento pois uma série de detalhes do XML já são interpretados automaticamente pelo próprio Parser.

Por outro lado, foi acompanhado o desempenho do sistema, projetando a sua utilização em um ambiente de uma ampla rede de sensores, e o principal gargalo observado foi no que se refere ao desempenho do EXPAT no processo de extração dos dados. Uma possível solução seria uma análise detalhada dos processo executados pelo EXPAT para a sua eventual personalização. Sendo esta biblioteca *open-source,* é possível esta intervenção, o que traria como contribuição uma versão para aplicações que exigem desempenho, como é caso da utilização do XML na transmissão de dados de sistemas de monitoramento.

Comparativamente o EXPAT em relação a outros parsers é um dos que apresenta um melhor desempenho, é três vezes mais rápido que o Xerces (desenvolvido na plataforma Java). O seu desempenho somente é superado por parsers que utilizam XML binário. O que seria uma estratégia alternativa de melhoria de desempenho do sistema.

Quanto à melhoria de desempenho uma outra implementação que poderia ser desenvolvida seria a migração do protocolo TCP/IP, na transmissão dos dados, para o

protocolo UDP. Restringindo o TCP/IP as operações remotas que exigem maior segurança, como por exemplo: seleção e configuração remota dos NCAP's

O PHP/NuSOAP foi utilizado tanto no desenvolvimento da aplicação Servidor quanto Cliente para o desenvolvimento do Web Service, o resultado da sua utilização foi positivo por apresentar uma facilidade de implementação e de acréscimo de novos serviços como por exemplo controle de atuadores. Além do mais, o protocolo SOAP prevê recursos de detecção de falhas na solicitação do serviço, importante para operações críticas, bem como criptografia.

A utilização do sistema operacional Linux se mostrou vantajosa, principalmente no que se refere ao gerenciamento dos sockets estabelecidos na porta 8080. Enquanto que o sistema Windows para aplicações envolvendo estes tipos de aplicações apresenta dificuldades de implementação, principalmente no que se refere ao ambiente de desenvolvimento .NET.

7. CONCLUSÕES E PESPECTIVAS FUTURAS

A integração dos computadores no ambiente da instrumentação permitiu que os avanços e tendências da informática começassem a ter uma forte influência também na instrumentação. A Internet é um exemplo deste fenômeno. Paralelamente, os avanços da microeletrônica contribuíram para o surgimento de microcontroladores e sensores mais sofisticados, tornando convidativo o emprego desses dispositivos em sistemas de instrumentação. No momento atual há uma grande associação destas duas tendências. Por outro lado a diversidade de protocolos e plataformas dificulta a interoperabilidade.

Como apresentado neste trabalho, uma complexidade a mais é acrescentada a estes desafios no que se refere às redes de sensores sem fio, pois a facilidade de alteração dos componentes desta rede gera uma dificulade extra na identificação das reconfigurações realizadas. Nesse sentido uma solução é integrar as arquiteturas e protocolos de uma rede de sensores ao conceito de sensores inteligentes, ou sensores *plug-and-play*, como é previsto pelas normas IEEE 1451.

Neste contexto foi desenvolvido um sistema remoto de monitoramento e aquisição de dados via Internet. Capaz de selecionar e configurar um conjunto de NCAP's, através de um Web Service. Todo o desenvolvimento deste sistema foi baseado em tecnologias de software livre e protocolos abertos.

A partir desses resultados surgem de imediato algumas perspectivas de trabalhos futuros:

- O mesmo sistema permite uma fácil migração para unidades remotas de telemetria. Bem como para sistemas de aquisição de dados associados a tecnologias de comunicação sem fio via telefonia celular, bastando a utilização da implementação JDDAC, na plataforma Java 2 Microedition;
- Associação de uma base de dados à interface de supervisão que permita a criação de contas de usuários, para o controle do acesso bem como para o registro dos dados coletados associados à sessão, de maneira que o sistema possa ser acessado simultaneamente por vários usuários;
- Recentemente foi disponibilizada para o JDDAC uma interface para a associação de transdutores reais ao sistema. Esta interface conhecida como

Transducer Interface Module (TIM) permite a comunicação com o sensor e leitura do TEDS, tendo como padrão de comunicação o protocolo 1-Wire, conforme previsto pela norma IEEE 1451. Esta expansão do sistema poderá ser implementada sem que sejam necessárias alterações das implementações realizadas nesta trabalho.

8. LISTA DE REFERÊNCIAS

AGILENT TECHNOLOGIES. Apresenta as definições e especificações da plataforma Agilent VEE. Disponível em:
< http://www.home.agilent.com/BRpor/nav/-536896708.0/pc.html>
Acesso em: 15 fev. 2005

Al-Ali, A.R. et al. Wireless smart sensors networks overview. Second IFIP International Conference on Wireless and Optical Communications Networks, 2005. Proceedings, March 2005, p. 536 - 540.

ARRUDA JR., C.R.E. **Context Kernel: um Web Service baseado nas dimensões de informação de contexto.** 2003. 85p. Dissertação (Mestrado) – Instituto de Ciências Matemáticas e de Computação da Universidade de São Paulo. São Carlos, 2003.

AYALA, D. et al. **Professional Open Source Web Services**. New York : Wrox, 2003. p. 305-328

BORGES, A. P. **Instrumentação virtual aplicada a um laboratório com acesso pela Internet**. 2002. 95p. Dissertação (Mestrado) – Escola Politécnica da Universidade de São Paulo. Departamento de Sistemas Eletrônicos. São Paulo, 2002.

CALVEZ, J. P. **Embedded real-time systems** New York : J. Wiley, 1993. p. 647

CAMPBELL, S.D. Web Services with NuSOAP.
Disponível em:
< http://www.zend.com/zend/tut/tutorial-campbell.php>.
Acesso em: 15 de dez. 2003.

CHESTER, T. Cross-platform integration with XML and SOAP*,* **IEEE IT Professional**, v.3, n.5, p.26 – 34, 2001.

CHONG, C. Y.; Kumar, S.P. Sensor networks: evolution, opportunities, and challenges. **Proceedings of the IEEE,** v. 91, n. 8, p. 1247 - 1256, 2003.

CONWAY, P. et al. IEEE 1451.2: An interpretation and example implementation. 17TH IEEE INSTRUMENTATION AND MEASUREMENT TECHNOLOGY CONFERENCE, 2000. Proceedings, v. 2, May 2000, p. 535 - 540.

CURBERA, F. et al. Unraveling the Web services web: an introduction to SOAP, WSDL, and UDDI. **IEEE Internet Computing**, v.6, n.2, p. 86-93, 2002.

EGAN, D**.;** The emergence of ZigBee in building automation and industrial control **Computing & Control Engineering Journal.** v. 16, n. 2, p. 14 - 19, 2005.

FERLINE, O. P. **Interconexão de redes Bluetooth - Uma aplicação em telemetria de serviços de distribuição de energia**. 2003. 172p. Dissertação (Mestrado) – Pontifícia Universidade Católica do Paraná. Programa de Pós-Graduação em Informática Aplicada. Curitiba, 2003.

FERRER, C.; Oliver, M. Overview and capacity of the GPRS (General Packet Radio Service). THE NINTH IEEE INTERNATIONAL SYMPOSIUM ON PERSONAL, INDOOR AND MOBILE RADIO COMMUNICATIONS, 1998. Proceedings, Sept. 1998, p. 106 - 110.

FERRIGNO, L.; et al. A Bluetooth-based proposal of instrument wireless interface. **IEEE Transactions on Instrumentation and Measurement**, v. 54, n. 1, p. 163 - 170, 2005.

GODFREY, K. The ever-changing face of integrated microprocessors-an evolutionary story. **Computing & Control Engineering Journal**, v.7, n.3, p. 153 – 160, 1996.

GUNZER, H. White paper: Introduction to Web Services. Borland – Março, 2002

HICKS, D. The Museum of HP Calculators. Disponível em:
< http://www.hpmuseum.org/>
Acesso em: 14 fev. 2005

HILL, J. et al. The platforms enabling wireless sensor networks. **Communications of the ACM,** v. 47, n. 6, p. 41-46, 2004

HORAN, B. The use of capability descriptions in a wireless transducer network. Disponível em: < http:// research.sun.com/techrep/2005/abstract-131.html />
Acesso em: 15 mar. 2005

HSIEH, T. T. Using sensor networks for highway and traffic applications. **Potentials, IEEE.** v.23, n.2, p. 13-16, 2004.

IEEE. IEEE standard codes, formats, protocols, and common commands for use with IEEE Std 488.1-1987, IEEE standard digital interface for programmable instrumentation. IEEE Std 488.2-1992. Dec. 1992

IEEE. IEEE standard for a smart transducer interface for sensors and actuators - transducer to microprocessor communication protocols and Transducer Electronic Data Sheet (TEDS) formats. IEEE Std 1451.2-1997. Sept. 1998.

IEEE. IEEE standard for a smart transducer interface for sensors and actuators - Network Capable Application Processor (NCAP) Information Model. IEEE Std 1451.1-1999. Apr. 2000.

IEEE. IEEE Standard for a Smart Transducer Interface for Sensors and Actuators-Digital Communication and Transducer Electronic Data Sheet (TEDS) Formats for Distributed Multidrop Systems. IEEE Std 1451.3-2003. Apr. 2004

IEEE. IEEE Standard for A Smart Transducer Interface for Sensors and Actuators - Mixed-Mode Communication Protocols and Transducer Electronic Data Sheet (TEDS) Formats. IEEE Std 1451.4-2004. Mar. 2005

JAMMES, F.; Smit, H. Service-Oriented Paradigms in Industrial Automation. **IEEE Transactions on Industrial Informatics,** v. 1, n. 1, p. 62 - 70 , 2005.

JDDAC. Downloads e especificações da plataforma.
Disponível em: <https://jddac.dev.java.net/>
Acesso em: 20 Mar. 2005

JIANG, Q. ; Manivannan, D. Routing protocols for sensor networks. FIRST IEEE CONSUMER COMMUNICATIONS AND NETWORKING CONFERENCE, 2004. Proceedings, Jan. 2004, p. 93 - 98.

JOUVRAY, C. et al. Smart sensor modeling with the UML for real-time embedded applications. 2004 IEEE INTELLIGENT VEHICLES SYMPOSIUM, 2004. Proceedings, June 2004, p. 919 - 924.

KANG, L.; Song, E. Object-oriented application framework for IEEE 1451.1 standard [smart sensors]. 21ST IEEE INSTRUMENTATION AND MEASUREMENT TECHNOLOGY CONFERENCE, 2004. Proceedings, May 2004, v. 2, p.1182 - 1187.

KAHN, J. M. et al. Mobile networking for smart dust. ACM/IEEE INTERNATIONAL CONFERENCE ON MOBILE COMPUTING AND NETWORKING, 1999. Proceedings, 1999 p. 271–278.

LEE, K.B.; Schneeman, R.D. Internet-based distributed measurement and control applications. **IEEE Instrumentation & Measurement Magazine**. v. 2, n. 2, p. 23 – 27, 1999.

LEE, K. C. et al. IEEE-1451-based smart module for in-vehicle networking systems of intelligent vehicles. **IEEE Transactions on Industrial Electronics**. v. 51, n. 6, p. 1150 - 1158, 2004.

MALAN, D. et al. CodeBlue: An Ad Hoc Sensor Network Infrastructure for Emergency Medical Care. WORKSHOP ON WEARABLE AND IMPLANTABLE BODY SENSOR NETWORKS, 2004. Proceedings, Sept. 2004. Disponível em: < http://www.eecs.harvard.edu/~mdw/papers/codeblue-bsn04.pdf> Acesso em: 20 fev. 2005.

MARINO, P.; Nogueira, J.; Hernandez, H. Laboratory of virtual instrumentation for industrial electronics. IEEE INTERNATIONAL CONFERENCE ON INDUSTRIAL TECHNOLOGY 2000. Proceedings, v. 1, Jan. 2000, p. 249 – 253.

MATEUS, G. R. ; LOUREIRO, A. A. F. Introdução a computação móvel. II ESCOLA DE COMPUTAÇÃO MÓVEL. Rio de Janeiro, 1998.

MCCLANAHAN, A. SCADA and IP: is network convergence really here? **IEEE Industry Applications Magazine**. v.9, n.2, p.29-36, 2003.

NATIONAL INSTRUMENTS. Apresenta as definições e especificações da plataforma LabVIEW. Disponível em:
< http://www.ni.com/labview/>
Acesso em: 15 fev. 2005.

NEMEROFF, J. et al. Application of sensor network communications. MILITARY COMMUNICATIONS CONFERENCE, 2001. IEEE Proceedings, 2001. p. 336 – 341.

NIST. Apresenta informações sobre as normas IEEE 1451 e exemplos de aplicação Disponível em:
<http://ieee1451.nist.gov/>
Acesso em: 16 fev. 2005.

OKAMOTO, Y. et al. The nature of the 21st century paradigm shift driven by the next-generation Internet. ENGINEERING MANAGEMENT SOCIETY, 2000. **IEEE Meeting**. Proceedings, 2000. p. 464 – 469.

OLIVEIRA, A. L. **Instrumentação virtual aplicada à caracterização de matrizes neuro-eletrônicas em microcavidades**. 1999. 60p. Trabalho de Formatura – Escola Politécnica da Universidade de São Paulo. Departamento de Sistemas Eletrônicos. São Paulo, 1999.

ONO, E. T. **Implantação de rede wireless de alta velocidade**. 2004. 108p. Trabalho de Formatura - Universidade Federal de Santa Catarina - Ciências da Computação. Florianópolis, 2004.

QIJUN, C. et al. Research on and pure Java realization of a Web-based mobile robot system. AMERICAN CONTROL CONFERENCE, 2003. Proceedings, 2003. V.1, p. 615 - 620

PUPO, M.S. **Interface homem-máquina para supervisão de um CLP em controle de processos através da WWW.** 2002. 83p. Dissertação (Mestrado) – Escola de Engenharia de São Carlos da Universidade de São Paulo. São Carlos, 2002.

RABBIT SEMICONDUCTOR. Informações técnicas sobre o microcontrolador Rabbit. Disponível em:
<http://www.rabbitsemiconductor.com/>
Acesso em: 16 fev. 2005.

REIS, C. P. **Embedded Systems Applications**. Universidade Católica de Pelotas, 2002. (Artigo técnico). Disponível em:
<http://atlas.ucpel.tche.br/~barbosa/consiso/consiso_6/art_so1_124/embarcado.pdf >
Acesso em: 01 mar. 2004.

SCHLEEF, D.; Hess, F. M. Documentação e download da plataforma Comedi.
Disponível em:
< http://www.comedi.org/>
Acesso em: 15 fev.2005

SHAHEEN, M. et al. Remote laboratory experimentation. AMERICAN CONTROL CONFERENCE, 1998. Proceedings, 1998. V. 2, p. 1326 - 1329.

SOHRABI, K. et al. Protocols for self-organization of a wireless sensor network**. IEEE Personal Communications.** v.7, n.5, p. 16-27, 2000.

SUZUKI, T et al. Teleoperation of multiple robots through the Internet. 5TH IEEE INTERNATIONAL WORKSHOP ON ROBOT AND HUMAN COMMUNICATION, 1996. Proceedings, 1996, p. 84 - 89.

SVEDA, M. A design framework for Internet-based embedded distributed systems. 11TH IEEE CONFERENCE AND WORKSHOP ON THE INTERNATIONAL ENGINEERING OF COMPUTER-BASED SYSTEMS, 2004. Proceedings, 2004, p. 113 - 120.

SZEWCZYK, R. et al. Habitat monitoring with sensor networks. **Communications of the ACM,** v. 47, n. 6, p. 34-40, 2004

TILAK, S. et al. A taxonomy of wireless micro-sensor network models. **ACM SIGMOBILE Mobile Computing and Communications Review,** v. 6, n. 2, p. 28-36, 2002

TI-YEN, Y.; Wolf, W. Performance estimation for real-time distributed embedded systems. **IEEE Transactions on Parallel and Distributed Systems**, v.9, n.11, p.1125 – 1136, 1998.

UPnP FORUM. Repositório de dados e forum de discussão sobre esta tecnologia. Disponível em: <http://www.upnp.org/about/default.asp>
Acesso em: 02 mar. 2005

VIEGAS, V. M. R. **Projeto e implementação de um sistema de sensores inteligentes baseado na norma IEEE 1451.** 2003. 241p. Dissertação (Mestrado) - Instituto Superior Técnico da Universidade Técnica de Lisboa. Lisboa, 2003.

VIEIRA, M. A. M. et al. Survey on wireless sensor network devices. EMERGING TECHNOLOGIES AND FACTORY AUTOMATION, 2003. Proceedings, 2003. p. 537-544.

VOGELS, W. Web services are not distributed objects. **IEEE Internet Computing**. v. 7, n. 6, p. 59 – 66, 2003.

WANT, R. Enabling ubiquitous sensing with RFID. **IEEE Computer.**V. 37, n. 4, p. 84 - 86, 2004.

WEISER, M. The computer for the 21st century. **ACM Mobile Computing and Communications Review**. V. 3, n. 3, p. 3 – 11, 1999.

WORLD WIDE WEB CONSORTIUM (W3C). Cambridge, MA, EUA. Apresenta as definições e especificações da arquitetura Web Services. Disponível em: < http://www.w3.org/TR/2004/NOTE-ws-arch-20040211/>.
Acesso em: 01 de fev. 2004a.

WORLD WIDE WEB CONSORTIUM (W3C). Cambridge, MA, EUA. Apresenta as definições e especificações do XML. Disponível em: <http://www.w3c.org/XML/>.
Acesso em: 01 de fev. 2004b.

WUNNAYA, S.V.; Hoo, P. Remote instrumentation access and control (RIAC) through inter-networking. SOUTHEASTCON '99. Proceedings IEEE, Mar. 1999, p. 116 – 121.

YARVIS, M.D. et al. Real-world experiences with an interactive ad hoc sensor network. INTERNATIONAL CONFERENCE ON PARALLEL PROCESSING WORKSHOPS, 2002. Proceedings IEEE, Aug. 2002, p. 143 - 151.

ZIGBEE ALLIANCE. Apresenta as definições e especificações da plataforma Zigbee. Disponível em:
<http://www.zigbee.org>
Acesso em: 21 fev. 2005