UNIVERSIDADE ESTADUAL DO SUDOESTE DA BAHIA – UESB
DEPARTAMENTO DE CIÊNCIAS EXATAS E TECNOLÓGICAS – DCET
COLEGIADO DO CURSO DE CIÊNCIA DA COMPUTAÇÃO

HELBER HENRIQUE LOPES MARINHO

SERVIDOR PARA GERENCIAMENTO REMOTO DE MICROCONTROLADOR ARDUINO EM LABORATÓRIO REMOTO

VITÓRIA DA CONQUISTA - BA

2015

HELBER HENRIQUE LOPES MARINHO

SERVIDOR PARA GERENCIAMENTO REMOTO DE MICROCONTROLADOR ARDUINO EM LABORATÓRIO REMOTO

Trabalho de conclusão de curso, para aprovação na disciplina Projeto Supervisionado II e como requisito parcial para obtenção do título de Bacharel em Ciência da Computação, na Universidade Estadual do Sudoeste da Bahia – UESB
Orientadora: Prof.ª Dra. Alzira Ferreira da Silva
Co-orientadora: Prof.ª M.ª Maísa Soares dos Santos Lopes

VITÓRIA DA CONQUISTA - BA

2015

“Que os vossos esforços desafiem as impossibilidades, lembrai-vos de que as grandes coisas do homem foram conquistadas do que parecia impossível. ”

(Charles Chaplin)

Resumo

Este trabalho apresenta a proposta de desenvolvimento de um software para um servidor de laboratório remoto a fim de solucionar um problema de comunicação entre o Servidor Web e um Arduino alocado em um robô autodirigido, através da Internet. Para isso, o software deve fornecer recursos para permitir que o usuário do sistema possa enviar e compilar código para Arduino remotamente, como também permitir a troca de dados pela interface de comunicação serial do Arduino.

Para o desenvolvimento da ferramenta, foi utilizada a linguagem de programação Java, apoiada por uma API para comunicação serial e arquivos *shellscript* do Linux. Foi seguida uma adaptação da metodologia de desenvolvimento de software XPlus para implementar e levantar os requisitos do sistema. Por fim, foram realizados testes de unidade e integração sobre o software desenvolvido.

Palavras-chave: Laboratório Remoto, Comunicação Serial, Internet, Arduino, XPlus.

Abstract

This work presents the proposal to develop a software for a remote lab server in order to solve a communication problem between the Web server and an Arduino allocated in a self-directed robot, via the Internet. For this, the software must provide resources to allow the system's user to send and compile Arduino code remotely, but also allow the exchange of data over Arduino's serial communication interface.

For the development of the tool, it was used Java programming language, supported by an API for serial communication and shellscript files from Linux. An adaptation of XPlus software development methodology was adopted to implement and define the system requirements. Finally, unit and integration tests were performed over the developed software.

Keywords: Remote Laboratory, Serial Communication, Internet, Arduino, XPlus.

SUMÁRIO

1 Introdução

Vive-se um momento de mudanças com o atual desenvolvimento tecnológico, novas formas de disseminar o conhecimento entre professor e aluno além das aulas presenciais estão surgindo a todo momento. Um exemplo disso é a prática de Ensino a Distância (EAD) que algumas instituições de ensino adotam para permitir ao aluno assistir aulas de qualquer lugar e a qualquer momento. Isso dá liberdade ao aluno de aprender de acordo com o seu ritmo de estudo e nos horários que melhor convém.

Neste contexto, torna-se interessante aplicar o conceito de EAD em laboratórios de experimentos práticos da instituição de ensino, criando um sistema de laboratórios remotos. E, desse modo, é dado ao aluno a oportunidade de colocar em prática aquilo que foi ensinado na teoria em sala de aula, como também, democratizando o acesso aos recursos de aprendizagem oferecidos pela instituição de ensino.

Os vários laboratórios remotos estudados apresentam uma arquitetura em comum: todos possuem uma interface de controle web, um servidor de laboratório e o equipamento ou maquete experimental. Como resultado, estes laboratórios especificam superficialmente o funcionamento de cada item da arquitetura, além de implementar soluções difíceis de adaptar em outros tipos de experimentos (Chella, 2007).

A partir desta constatação e em conjunto com o projeto de pesquisa LARA (Laboratório Remoto em AVA), tornou-se necessário desenvolver um software servidor de laboratório remoto para controle de dispositivos robóticos que utilizam a plataforma Arduino como microcontrolador e que permita o envio e compilação de código Arduino escrito pelo usuário de forma autônoma através da Internet, dispensando presença física no laboratório.

Para o desenvolvimento deste trabalho foi utilizado uma adaptação da metodologia de desenvolvimento de software XPlus, que é uma metodologia que integra o XP (eXtreme Programming) com práticas de projeto de interface voltado para experiência do usuário. Sendo o sistema modelado em diagramas de caso de uso e implementado na linguagem de programação Java.

Os objetivos definidos neste trabalho foram divididos em geral e específico. O objetivo geral foi desenvolver um servidor para gerenciamento remoto da plataforma Arduino a partir da Internet para o LARA. E os objetivos específicos, foram:

a. Identificar os requisitos funcionais e não funcionais do sistema de controle;

b. Estabelecer a arquitetura do sistema que permita implementação dos requisitos identificados;
c. Implementar o sistema de controle para gerenciamento da plataforma Arduino remotamente;
d. Avaliar o sistema implementado quanto à satisfação dos requisitos através de teste de softwares.

O sistema implementado deve proporcionar ao usuário, através da Internet, os seguintes serviços:

- Permitir a compilação de código Arduino;
- Permitir o envio de código ao Arduino;
- Retornar ao usuário as informações do compilador após a compilação ou envio de código;
- Permitir que o usuário envie dados até à interface de comunicação serial do Arduino, além de permitir que o usuário receba dados da interface de comunicação serial do Arduino.

Durante a fase de pesquisa, a maior dificuldade foi não ter encontrado materiais bibliográficos diretamente relacionados à proposta deste trabalho. Encontrando apenas trabalhos semelhantes que pouco exploravam os objetivos aqui definidos. Deduz-se que esta dificuldade vem por ter objetivos específicos ainda pouco explorados em sistemas de laboratórios remotos, assim como, não são publicados materiais que mostrem a implementação de um servidor de laboratório remoto. Portanto, o software desenvolvido neste trabalho se configura como uma pesquisa aplicada ou tecnológica, pois visa desenvolvimento de um sistema de comunicação que promove o controle do microcontrolador Arduino remotamente por meio de uma conexão com a Internet, permitindo que o usuário através de um navegador web possa acessar os recursos do sistema de laboratório remoto.

Este trabalho está dividido em 5 capítulos. No segundo capítulo, são descritos os conceitos de laboratório remoto, Arduino e de alguns meios de comunicação entre dispositivos eletrônicos. O terceiro capítulo, é feita a apresentação de alguns trabalhos relacionados encontrados durante a pesquisa bibliográfica, com uma breve descrição de cada um. No quarto capítulo é descrito o ambiente em que o sistema foi desenvolvido, os materiais utilizados e os requisitos de funcionamento. Também é descrito a metodologia de software, suas etapas de desenvolvimento, implementação do servidor

de laboratório e, por fim, os testes realizados. No quinto e último capítulo, apresentam-se as conclusões sobre todo o trabalho e apontamentos para trabalhos futuros.

2 Referencial Teórico

Este capítulo aborda conceitos sobre Laboratórios Remotos, Controle de Dispositivos Remotos e Desenvolvimento de Sistemas Embarcados, estes conceitos têm fundamental importância para compreender o desenvolvimento da solução apresentada.

2.1 Laboratório Remoto

O controle de dispositivos à distância está cada vez mais tangível nas indústrias, em equipamentos médicos, operações militares, na exploração espacial, entre outros. Direcionando, assim, a um novo paradigma para solução e implementação de sistemas diversos. Neste contexto, temos o crescente interesse de professores e pesquisadores em praticar atividades remotas com estudantes, utilizando de laboratórios com acesso remoto, dando ao estudante a oportunidade de colocar em prática aquilo que foi ensinado em teoria em sala de aula, como também, democratizando o acesso aos recursos de aprendizagem oferecidos pela instituição de ensino.

Cruz (2009) relata que

> Os laboratórios remotos despontam como uma nova possibilidade de acesso a equipamentos didáticos para o ensino, onde o seu controle é disponibilizado pela Internet para a realização de ensaios, permitindo que um número maior de estudantes possa ter acesso a esses instrumentos. Câmeras de vídeo podem ser adicionadas ao ambiente do laboratório de modo que o usuário remoto consiga ter uma visão e acompanhar de modo online, as ações que ele programa para serem executadas no laboratório real.

E o autor defende o seu uso ao dizer que “os laboratórios remotos despontam como uma nova possibilidade de acesso a equipamentos didáticos para o ensino, onde o seu controle é disponibilizado pela Internet para a realização de ensaios, permitindo que um número maior de estudantes possa ter acesso a esses instrumentos”.

Um sistema de laboratório remoto tem como vantagens permitir o acesso agendado pelo estudante, realizar testes a fim de obter resultados reais de um dado teórico, a execução de atividades planejadas pelo professor, assim como, permitir que o professor acompanhe o desempenho do estudante na execução de suas atividades.

Usar um navegador web padrão como interface do laboratório remoto, traz muitas vantagens em conexões via Internet, uma vez que, o uso de software adicional no computador do cliente é mínimo. Por outro lado, pode-se ter a dificuldade em adaptar

parâmetros de controle em sua interface gráfica, como também um modo de devolver as respostas do sistema.

2.2 Arduino

Para implementação deste trabalho foi utilizado a plataforma Arduino que é um dispositivo computadorizado de baixo processamento que tem a capacidade de se conectar a sensores e atuadores para interagir com o mundo real. De acordo com informações em seu site oficial[1], sua plataforma é de código livre utilizando um chip microcontrolador de baixo custo, com inúmeras possibilidades de utilização. Possui um ambiente de desenvolvimento, também de código livre, para escrever o software que irá dar funcionalidade ao dispositivo. O seu surgimento aconteceu em 2005 na cidade de Ivrea, Itália, com o objetivo de oferecer às escolas locais uma plataforma de prototipagem de baixo custo.

A capacidade do Arduino em desenvolver interatividade com objetos, obter dados dos mais variados tipos de sensores e botões, ligar luzes, controlar motores e quaisquer outros dispositivos físicos, como também a possibilidade de realizar projetos de tomada de decisão autônomos ou que utilizem da comunicação com um computador, traz inúmeras aplicações para professores, estudantes e entusiastas em eletrônica.

Podemos considerar as seguintes vantagens para utilizar a plataforma Arduino:

- Baixo custo - As placas possuem um custo menor quando comparado a outros microcontroladores no mercado. Seus valores vão desde o chip Arduino para ser montado pelo usuário final, até a placa já fabricada que custa menos de $50 (dólares americanos).
- Multi-Plataforma - O ambiente de desenvolvimento Arduino, ao contrário de outros microcontroladores, tem compatibilidade com a maioria dos sistemas operacionais disponíveis no mercado como Windows, Macintosh OSX e as mais diversas distribuições do Linux, além de ser compatível com outros tipos de computadores, como alguns smartphones que possui o sistema operacional Android.
- Ambiente de desenvolvimento simplificado - O Arduino IDE (do inglês *Integrated Development Environment*, em tradução livre Ambiente Integrado de Desenvolvimento), é fácil de usar para iniciantes e bastante flexível para usuários avançados realizarem suas tarefas. Em relação ao uso acadêmico, o fato do Arduino IDE possuir interface gráfica bastante similar às outras IDE de programação, tornando mais fácil o processo de ensino e aprendizagem para

[1] https://www.arduino.cc/en/Guide/Introduction (Acessado em 08/12/2014)

professores e alunos, respectivamente. Em suas primeiras versões disponíveis, o IDE do Arduino apenas permitia que o código fosse compilado e enviado para o microcontrolador através de sua interface gráfica no computador em que está instalado. A partir da versão 1.5.2 BETA[2] tornou-se possível realizar estes mesmos processos por comandos utilizando o console do sistema operacional.

Todo dispositivo Arduino possui pelo menos uma porta serial que fornece um meio de comunicação com outros dispositivos tais como módulos Bluetooth, ZigBee e com o computador, além de também ser a porta de comunicação para o envio de código pelo IDE do Arduino. A comunicação ocorre através dos pinos digitais 0 e 1 do Arduino, os mesmos utilizados pela conexão USB do dispositivo.[3]

A biblioteca padrão do Arduino oferece a classe *Serial*, que encapsula algumas funcionalidades para promover a comunicação serial, a fim de facilitar o seu uso. Segue algumas das funções disponíveis pela classe:

- begin(): esta função habilita a comunicação serial e recebe como parâmetro a velocidade da conexão em símbolos por segundo (*bauds*);
- print(): esta função recebe como parâmetro qualquer tipo de dados padrão (*char*, *int*, *long*, *float*, etc) e escreve em ASCII na porta serial;
- println(): com o comportamento semelhante à função *print()*, esta apenas acrescenta a quebra de linha ao final do envio dos dados;
- read(): assim que algum byte de dado chega ao Arduino, ele é armazenado num buffer de 64 bytes e, através desta função é possível fazer a leitura dos dados recebidos sem a utilização de parâmetro.

2.3 Processo de comunicação entre dispositivos eletrônicos

O método de comunicação do computador com um dispositivo eletrônico varia de acordo com a tecnologia empregada no processo. Tecnologias que utilizam meios cabeados tendem a obter maiores velocidades na comunicação, já tecnologias que dispensam o uso de fios oferecem maior liberdade ao projeto de sua aplicação. A seguir serão descritos alguns meios de comunicação com suas vantagens e desvantagens e, também, será descrito o processo de comunicação utilizando a Internet.

[2] https://www.arduino.cc/en/Main/ReleaseNotes (Acessado em 16/04/2015)
[3] http://playground.arduino.cc/Referencia/Serial (Acessado em 17/04/2015).

2.3.1 Comunicação Serial

A comunicação serial é bastante utilizada pela indústria eletroeletrônica por ser o padrão de comunicação entre hardwares mais simples, sendo na maioria das vezes reconhecida pelo padrão RS-232 (Recommended Standard). Sua principal característica está no fato de estabelecer comunicação entre dispositivos com a utilização de apenas 3 fios: Rx (recebimento de dados), Tx (transmissão de dados) e GND (comum).

A comunicação serial está presente na maioria dos microcontroladores, possuindo, em grande parte, um módulo exclusivo para esta comunicação RS-232 (Numajiri, 2003). Dessa forma, torna-se simples e prático criar o processo de comunicação entre o computador e o dispositivo microcontrolador.

Além dos computadores e microcontroladores, vários módulos e sensores também utilizam desta comunicação para transmitir dados, dentre eles estão o Blueooth e ZigBee que serão descritos nos tópicos 2.3.2 e 2.3.3.

2.3.2 Bluetooth

Bluetooth é um padrão de comunicação sem fio de curto alcance, que opera na frequência ISM (*Industrial, Scientificand Medical)* de 2,4 GHz, a qual não necessita de autorização para utilizá-la na maioria dos países. Seu nome veio do Viking: *Harald Blätand* (*Harald the Bluetooth*) que conseguiu unir reinos distintos em meio a conflitos de territórios e religiões que, na época, dividia a Europa. De modo análogo, esta tecnologia tem a capacidade de unir diferentes dispositivos eletrônicos sem complicações. Suas principais características são robustez, baixa potência e baixo custo.

De acordo com Kobayashi (2004), os dispositivos Bluetooth conectam-se através de *piconets*, que são, pequenas redes que possuem um dispositivo mestre capaz de conectar a até sete dispositivos escravos (*slaves*) e utilizam a mesma *hoppingsequence*[4], no raio de cobertura de dez metros. A *hoppingSequence* do canal é definida com base no endereço mestre que definiu a *piconet* (Fernandes, 2012). Todos os dispositivos conectados ao *piconet* têm seus relógios sincronizados ao relógio do

[4] HoppingSequence é a alternância periódica e sincronizada nos canais de comunicação entre os dispositivos Bluetooth para evitar interferências com outras tecnologias sem fio que operam na faixa de 2,4 GHz, além de aumentar a segurança na comunicação por estar sempre trocando o canal de comunicação (Albuquerque, 2010).

mestre. Cada dispositivo deve solicitar ao mestre permissão para comunicar com os outros.

O Bluetooth foi desenvolvido com o propósito de facilitar a conexão entre dispositivos com tecnologias distintas, usando tecnologia sem fios e vários protocolos que garantem a segurança e privacidade na transmissão e recepção de informações.

2.3.3 ZigBee

ZigBee é uma tecnologia desenvolvida pela ZigBee Alliance, uma associação com mais de 300 empresas em prol de estabelecer um padrão de comunicação para redes sem fio. Segundo Fernandes (2012), a tecnologia ZigBee trabalha de modo análogo às abelhas, que passa pelas flores e trocam informações com outras abelhas sobre onde encontrar recursos. Diferentemente de outros tipos de redes, o ZigBee não tem altas taxas de transferência de informações entre os dispositivos, priorizando sua comunicação apenas para sensores e dispositivos de controle. Dessa forma, esta tecnologia oferece a simplicidade e baixo custo de manutenção, a segurança no tráfego de informações, latência baixa, e uma maior economia de energia.

Assim como a tecnologia Bluetooth, o ZigBee opera na frequência ISM de 2,4 GHz (Global), 858 MHz (Europa) e 915 MHz (EUA e Austrália), com taxas de transmissão que variam entre 20 kbps e 250 kbps a depender da localidade do dispositivo e, pode suportar até, em teoria, 65.536 dispositivos numa mesma rede (Fernandes, 2012).

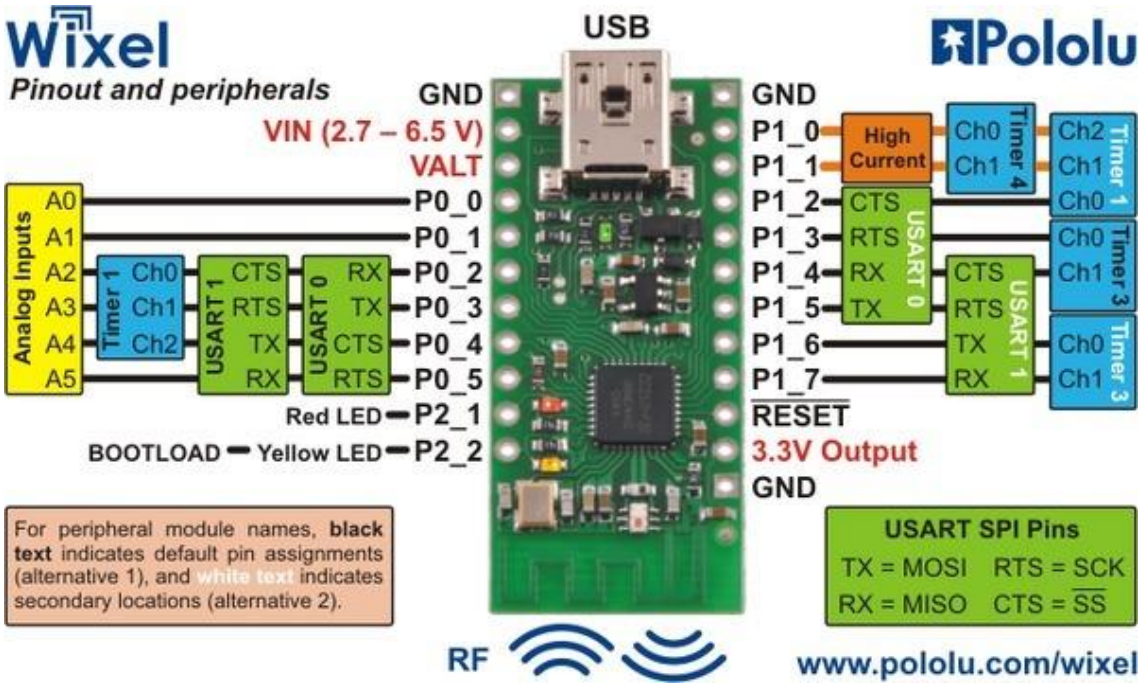
Uma característica importante da rede ZigBee está no fato de sua arquitetura permitir a estruturação em forma de malha, onde os dispositivos são fortemente interconectados, aumentando, assim, a confiabilidade, o alcance de cada dispositivo e a segurança da rede. Desse modo a rede se torna descentralizada e sempre que houver uma falha em um dispositivo, a rede logo irá se reestabelecer através de outros caminhos, garantindo a conformidade na comunicação.

2.3.4 Wixel Shield

O Wixel Shield é um módulo desenvolvido pela Pololu Robotics & Eletronics para comunicação serial sem fio compatível com a plataforma Arduino e computadores em geral.

A comunicação por conexão sem fio ocorre na frequência de 2,4 GHz e pela interface USB (*Universal Serial Bus*) conectado ao computador. O Wixel possui o microcontrolador CC2511F32 fabricado pela Texas Instruments, que tem integrado o transmissor e receptor via radio, 32 KB de memória flash, 4 KB de memória RAM, e interface USB *Full-Speed*[5]. A Figura 1 mostra os conectores e suas funcionalidades no Wixel.

Figura 1 - Esquema de conectores e conexões do Wixel.



Fonte: https://www.pololu.com/docs/0J46/1.a (Acessado em 25/06/2015)

Assim como na interface de comunicação serial, o Wixel também fornece as conexões Rx, Tx e GND, além de conexões como CTS e RTS que também podem ser encontradas em dispositivos de comunicação serial. Dessa forma, temos que o Wixel pode ser utilizado para substituir a conexão serial convencional no Arduino, por eliminar as limitações impostas pelos cabos ao projeto do laboratório remoto com um robô autodirigido.

[5] https://www.pololu.com/docs/0J46/1 (Acessado em 25/06/2015)

2.3.5 Internet

A Internet teve seu início na Arpanet, uma rede da *Advanced Research Projects Agency* (*ARPA*), que foi criada pelos militares americanos ao final da década de 1960 durante a disputa do poder mundial entre os Estados Unidos e a União das Repúblicas Socialistas Soviética, com o objetivo de desenvolver uma rede de computadores descentralizada, estável e flexível, que fosse capaz de continuar operando mesmo após a um ataque nuclear (Castells, 2003).

Segundo Tanenbaum (2003),

> A Internet não é de modo algum uma rede, mas sim um vasto conjunto de redes diferentes que utilizam certos protocolos comuns e fornecem determinados serviços comuns. É um sistema pouco usual no sentido de não ter sido planejado nem ser controlado por ninguém.

Para entender o funcionamento da Internet, temos que destacar, entre os vários, o protocolo TCP/IP (*Transfer Control Protocol/Internet Protocol*), que tem a função de conectar diferentes máquinas em diferentes redes sem a necessidade de adaptações nas máquinas e nas redes. Obtendo, assim, a intercomunicação entre todos os sistemas e, consequentemente, tornando a Internet como o meio de comunicação mais abrangente em todo o mundo.

Dessa forma, torna-se possível que as comunidades que foram excluídas dos meios de comunicação em massa possam alcançar a informação oferecida neste vasto mundo virtual repleto de ideias, culturas, conhecimento científicos e muito mais, em prol de um bem em comum. Sendo assim, podemos inferir que tais conhecimentos disponíveis a qualquer um podem ser usados para a construção de novos valores. Como também pode ser usado para o compartilhamento de conhecimentos.

Para Pretto (2013), a queda do preço dos computadores e a disseminação da computação entre os mais variados dispositivos portáteis, como *smartphones*, *tablets* e *notebooks*, possibilitou que o compartilhamento de informações sob a forma de vídeos, imagens, textos, sons, etc. possa ser alcançado por qualquer um e, consequentemente, pode ser repassado para qualquer um, aumentando a eficiência e descentralizando as fontes de informação.

Está claro que com o rápido crescimento da Internet e o grande avanço das tecnologias de comunicação em diversos dispositivos, o acesso à Internet pode ser obtido facilmente por todos. Logo, um sistema que utiliza esse meio de comunicação

para interagir com o usuário, irá garantir que seu uso chegue a qualquer um e em qualquer lugar.

Com a Internet cada vez mais presente no cotidiano fica claro que seu uso se tornou essencial no mundo contemporâneo. Seu uso pode ser visto em, praticamente, todos os lugares, desde o envio de uma simples mensagem de e-mail ou para a realização de uma transação bancária, ou até mesmo para informar a um médico, que está em sua casa, sobre o estado de um paciente que está no hospital. Porém, da mesma forma que a Internet pode ser usada para promover o convívio social, também pode ser usada por pessoas mal-intencionadas com o objetivo de roubar dados pessoais e bancários, ofender, discriminar, etc. Isso ainda ocorre devido às dificuldades encontradas pelos órgãos competentes para identificar e punir o usuário infrator por trás do computador.

2.4 Conclusões

Neste capítulo foram apresentados breves conceitos de alguns dos mais diversos meios de comunicação existentes. Dentre as tecnologias apresentadas, temos que o Bluetooth e ZigBee possuem protocolos bem definidos para trabalhar, principalmente, com alta segurança e com múltiplos dispositivos de mesma tecnologia. Porém demonstram grandes limitações quanto à sua aplicação em ambientes que possuem como requisitos comunicação a longa distância e disponibilidade da tecnologia para qualquer usuário do sistema.

Sendo assim, a Internet, comunicação serial e Wixel Shield melhor se adequam aos requisitos do sistema para construção do laboratório remoto. A Internet, por ser universalmente o principal meio de comunicação entre os computadores e dispositivos, se torna a melhor opção para disponibilizar o acesso ao sistema pelos usuários. E o Wixel Shield, que já engloba a interface de comunicação serial, dá liberdade para o objeto de experimento do laboratório remoto se locomover pelo espaço do laboratório físico, eliminando o uso de fios para a comunicação entre o Arduino e o computador servidor de laboratório.

3 Trabalhos relacionados

Martins (2002) infere que “adaptar-se a mudanças tecnológicas é necessário. Torna-se, assim, premente proporcionar aprendizagem autônoma, visando à formação de cidadãos responsáveis e democraticamente intervenientes na vida comunitária”.

Sendo assim, é relevante que proporcione ao estudante a autonomia em seu processo de aprendizagem, deixando-o aprimorar e obter novos conhecimentos por méritos próprios, para o crescimento responsável de seus princípios e condutas.

Dentro desse contexto, nesta seção são apresentados alguns trabalhos relacionados que incluem sistemas de laboratórios remotos para educação a distância e suas metodologias de desenvolvimento. Outros trabalhos não foram citados por não terem sido encontrados durante a pesquisa bibliográfica ou por não compreenderem os objetivos deste trabalho.

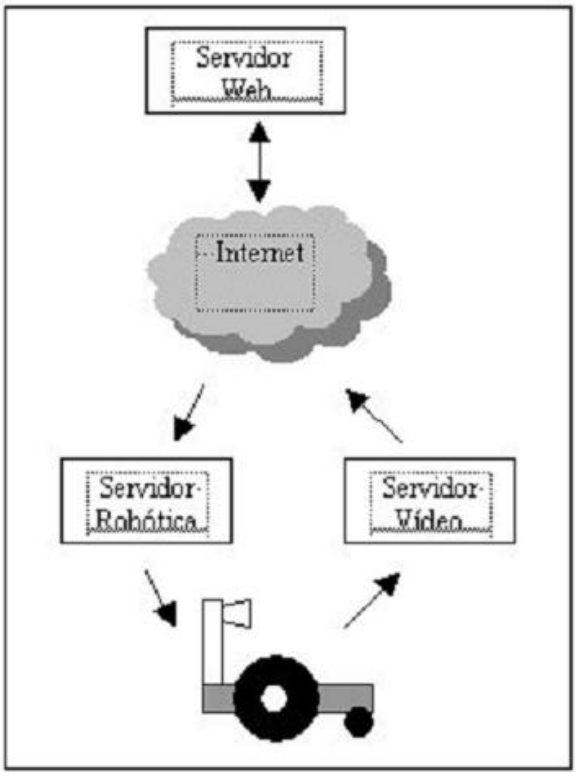
3.1 Robótica Educacional e Acesso Remoto: Relato de uma Experiência

Este trabalho foi desenvolvido no Núcleo de Informática Aplicada à Educação (NIED) da Universidade Estadual de Campinas (UNICAMP), sua proposta é inserir o computador como ferramenta educacional para a interação com os mais diversos dispositivos robóticos de finalidades pedagógicas (D’Abreu e Martins, 2007).

Com isso, foi desenvolvido o Robô Móvel utilizando a abordagem de Laboratório de Acesso Remoto. Constituído por um robô móvel dotado de uma câmera de vídeo, aplicativos para o controle remoto e operando no modelo cliente-servidor via Internet, e um servidor para codificação e transmissão de vídeo capturados pelo robô em tempo real. A Figura 2 fornece uma visão esquemática do sistema.

O controle remoto do robô ocorre através do acesso à uma página web em um navegador web padrão. A página de controle possui botões direcionais que enviam o comando de movimento ao robô, uma área de texto para anotações durante o experimento, um botão para captura de fotos e uma janela para visualização do vídeo do robô em tempo real. Em contrapartida, o sistema proposto neste trabalho dá liberdade ao usuário de escrever o código que irá programar o carrinho robotizado, além de fornecer o console de comunicação serial que permite o recebimento de informações adversas dos sensores, como também, o envio de comandos personalizados para o mesmo.

Figura 2 - Esquema do Laboratório de Acesso Remoto com Robô Móvel



Fonte: D'Abreu (2007).

Sua concepção se deu numa plataforma que oferecia confiabilidade, robustez e baixo custo. Em seu microcontrolador está conectado dois servomotores, que são responsáveis pela locomoção. E sua comunicação com o computador é feita por um link de rádio frequência conectado à porta paralela do computador.

3.2 Metodologia para Implantação de Laboratórios Remotos via Internet na Área de Automação da Manufatura

Este trabalho apresenta uma metodologia para implementação de Laboratórios Remotos via Internet voltado para o Ensino a Distância com experimentos remotos na área de automação da manufatura e robótica. Sua motivação se dá pelos benefícios que um software oferece ao demonstrar para os alunos conceitos abstratos de um determinado assunto.

De acordo com Álvares e Ferreira (2003), seu principal objetivo é disseminar o acesso a equipamentos presentes em instituições de ensino, mas que nem sempre está disponível para manuseio por estudantes, uma vez que, têm alto custo de aquisição.

Em sua metodologia é utilizada conexão com a Internet através dos protocolos TCP/IP. Foi necessário compactar a transmissão de vídeo para que pudesse transmitir ao usuário imagens das aplicações no laboratório em tempo real e de forma satisfatória. A interface gráfica foi projetada para ser construída em navegadores web, baseada nas linguagens de programação HTML, Javascript e Java (Álvares e Ferreira, 2003).

Este trabalho cita seis aplicações que foram desenvolvidas baseada na metodologia proposta. Os trabalhos são: RobWebCam, RobWebLink, Robô Móvel MRL 1.0, teleoperação do robô Nomad XR4000, teleoperação da máquina de Oxi-corte White Martins e, por fim, o sistema WebCAPP Variante, o que possibilitou a validação da metodologia.

Todas as aplicações desenvolvidas com a metodologia oferecem uma interface gráfica num navegador web padrão e suporte à transmissão de vídeo em tempo real da aplicação remota. Possuem também uma área de controle composta por diversos botões que variam de acordo com a finalidade da aplicação empregada. Porém, esta área de controle limita o usuário manipular a aplicação, permitindo-o interagir somente com as funcionalidades previamente disponibilizadas.

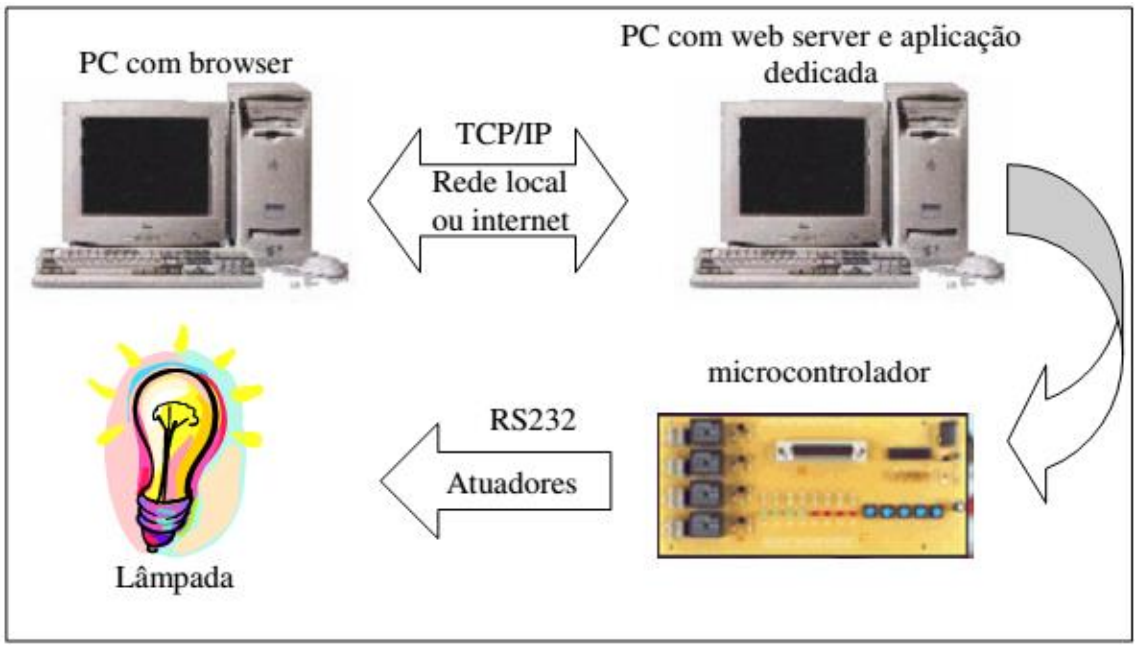
3.3 Desenvolvimento de um Sistema de Controle de Dispositivos Via Acesso Remoto

Em seu trabalho, Numajiri (2003) objetiva criar um modelo base de um servidor de controle de dispositivos para promover interação remota via Internet ou rede local, de modo que se faça necessário apenas pequenas adaptações em seu código. Um outro objetivo que Numajiri planeja alcançar é levar o projeto para o sistema operacional Linux, por ser um ambiente de código livre, gratuito e repleto de programas semelhantes ao do sistema operacional dominante no mercado de computadores pessoais, Windows.

Para o desenvolvimento do software, é necessário a criação de uma arquitetura de rede que ofereça suporte para os computadores conectados acessarem a interface de controle. Sendo assim, a arquitetura irá permitir que o computador remoto controle os dispositivos eletrônicos conectados ao computador. Como atividade complementar, está o estudo para implantação do sistema em navegadores web utilizando as linguagens *script* CGI (Common Gateway Interface), linguagem de programação C, e páginas web. A Figura 3 ilustra a arquitetura proposta para o sistema.

Com o modelo do sistema definido e implementado, a próxima fase que segue é o desenvolvimento de aplicações sobre esta arquitetura criada. Como o próprio Numajiri infere, esta fase não tem fim, visto que são inúmeras as possibilidades de aplicações que podem ser criadas baseando nas que já existem e nas que podem ser criadas com o surgimento de novas tecnologia.

Figura 3 - Arquitetura final do sistema proposto por Numajiri (2003)



Fonte: Numajiri (2003)

A tecnologia na área de microcontroladores teve um grande avanço nos últimos anos, o que facilitou, simplificou e permitiu a realização de atividades eletrônicas em geral. Possibilitou aos entusiastas em eletrônica realizarem projetos utilizando placas prontas para prototipagem com microcontroladores bastante flexíveis, poderosos e, principalmente, a um baixo custo. Na mesma forma que estão sendo oferecidos módulos de sensores e atuadores que complementam o desenvolvimento do pequeno projeto. Finalizando, hoje existem módulos dos microcontroladores que disponibilizam conexão direta à rede de computadores, tornando desnecessário o uso de um computador mediador para controlar o dispositivo remotamente e, até mesmo, para oferecer uma página web.

3.4 Conclusões

Durante a pesquisa bibliográfica não foram encontrados muitos trabalhos diretamente relacionados com os objetivos propostos neste trabalho. Entretanto, nos trabalhos encontrados a maioria têm a metodologia para criar um sistema de laboratório remoto com arquiteturas semelhantes que utilizam de um servidor de laboratório, servidor web, dispositivo microcontrolador, usuário acessa o sistema a partir de um navegador web padrão e, quando disponível, câmera de vídeo para o experimento. Este trabalho além de seguir o modelo desta arquitetura, irá focar na aplicação de servidor de laboratório que oferece a possibilidade de o usuário compilar e enviar o código Arduino para o microcontrolador remotamente, utilizando tecnologias gratuitas e/ou de código aberto.

4 Servidor de Laboratório LARA

Neste capítulo será descrito o processo de desenvolvimento do software para efetuar a compilação de código remotamente, estabelecer a conexão serial e enviar um novo código ao microcontrolador do Arduino através da Internet.

4.1 Descrição do Problema

Este trabalho surgiu da necessidade de se resolver o problema de comunicação entre uma interface web e um veículo autodirigido que é um experimento do laboratório remoto proposto para o LARA.

O LARA é um projeto de pesquisa desenvolvido no Laboratório de Sistemas Inteligentes Automação e Controle (SIAC) sobre coordenação do Curso de Ciências da Computação da Universidade Estadual do Sudoeste da Bahia – UESB. Este projeto tem como objetivo geral desenvolver um laboratório remoto de robótica integrado ao ambiente virtual de aprendizagem Moodle[6], baseado em tecnologias de código aberto para ensinar linguagem de programação, algoritmos, inteligência artificial e robótica educativa (Silva, 2014).

Um dos experimentos para o laboratório remoto é o controle de um robô autodirigido que deve ser acessado remotamente através de uma interface de programação e controle.

O veículo autodirigido presente no LARA foi construído por parte da equipe de pesquisadores e colaboradores do projeto, e buscou a utilização de materiais de baixo custo, mas que ainda mantivesse as características e padrões de qualidade exigidos na competição RoboCup Junior Rescue A. Aproximando, assim, os alunos à robótica e competições.

A base do robô é constituída por dois motores com uma roda cada, com tensão de trabalho de 5v ~ 6v, e por uma roda boba. Sendo essa a configuração mais popular para robôs em ambientes internos (Zepelin e Borda, 2012). Acoplado ao carro autodirigido, estão três sensores ultrassom para obter referência do espaço entre o robô e um obstáculo durante o percurso. Também estão três pares de sensores infravermelhos

[6] "O **Moodle** é uma plataforma de aprendizagem a distância baseada em software livre. É um acrônimo de Modular Object-Oriented Dynamic Learning Environment (ambiente modular de aprendizagem dinâmica orientada a objetos)" (Sabbatini, 2007).

(emissor e receptor) utilizados em exercícios que exige ao robô seguir linha impressa na arena. Por fim, tem-se o sistema de alimentação contínua que fornece 5v para os periféricos do robô (sensores e motores) e 12v para o Arduino.

Com o robô autodirigido definido, torna-se necessário desenvolver o código para o microcontrolador gerenciar os seus sensores e motores. Esta tarefa é normalmente realizada por um Ambiente de Desenvolvimento Integrado (do inglês Integrated Development Environment, ou IDE) de código aberto fornecida pelo Arduino, que tem como principais funcionalidades a compilação do código para verificar a existência de erros de sintaxe, envio do código escrito para o microcontrolador e monitor de comunicação serial. No laboratório remoto do projeto LARA, estas funcionalidades do IDE e acesso à câmera de experimento também deverão estar disponíveis na interface do usuário do sistema. Outros requisitos definidos no projeto são, ter funcionamento 24 horas durante todos os dias da semana e, para o seu desenvolvimento, utilizar preferencialmente softwares e hardware de código aberto e baixo custo.

Dentro deste contexto, este projeto se configura como uma pesquisa aplicada ou tecnológica, pois visa desenvolvimento de um sistema de comunicação que promova o controle do Arduino remotamente por meio de uma conexão com a Internet, permitindo que um navegador web possa acessar os recursos do sistema.

Como visto, tem-se o propósito de aplicar conhecimentos teóricos e procedimentos práticos para o desenvolvimento de um produto com finalidades bem práticas (Fuck e Vilha, 2011). Esta questão também é destacada por Vilaça (2010) que “a pesquisa aplicada tem como motivação a necessidade de produzir conhecimento para aplicação de seus resultados, com o objetivo de “contribuir para fins práticos, visando à solução mais ou menos”. O autor destaca, também, que “na maioria dos casos, as pesquisas aplicadas exigem e partem de estudos teóricos”.

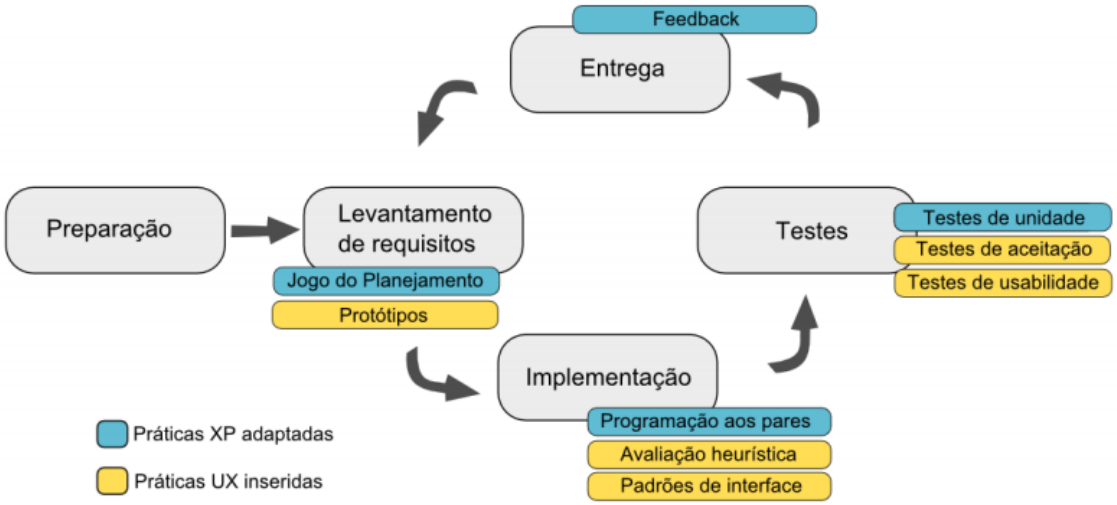
Desta forma, para este trabalho foram pesquisados processos de comunicação entre dispositivos eletrônicos, no caso, para sistema embarcado e metodologia de desenvolvimento de software que atenda a necessidade especificada no projeto. E, em seguida, para sua implementação foi criado o Servidor de Laboratório do LARA (SLL), utilizando o espaço cedido pela UESB, o SIAC.

Como para o bom desenvolvimento de software requer a utilização de uma metodologia de desenvolvimento, neste trabalho, para atender as características do projeto, foi aplicada uma adaptação da metodologia ágil de desenvolvimento XPlus e esta pôde ser empregada ao desenvolvimento de sistemas embarcados.

4.2 Descrição da Metodologia

O processo de desenvolvimento de software está cada vez mais complexo, o que antes poucas linhas de código e poucas funcionalidades eram o bastante para um cliente, hoje já não é mais. Os clientes atualmente estão mais exigentes e demandam por softwares completos, com eficiência e eficácia. Para atender essa demanda com baixo custo, alta qualidade e em tempo hábil, é necessário seguir uma metodologia de desenvolvimento que possui um conjunto de atividades e resultados esperados. Fazendo parte de etapas cuidadosamente planejadas, onde o sucesso de cada etapa é essencial para o correto funcionamento do produto final. A metodologia utilizada para o desenvolvimento do LARA foi uma adaptação do XPlus (Guimarães e Chaves, 2009), que é uma metodologia que integra o XP (eXtreme Programming) com práticas de projeto de interface voltado para experiência do usuário (User Experience ou UX). As boas práticas UX propõe que a interface deve ser mais que visualmente agradável, ela deve estar na forma que o usuário espere que seja. A interface deve se adaptar à realidade dos usuários e não o usuário que deve se adaptar à interface, pois, uma interface mal projetada para o usuário pode ocasionar a desistência na hora de utilizar o sistema e, com isso, prejuízos. A metodologia XPlus define cinco fases de desenvolvimento, como mostra a Figura 4.

Figura 4 - Visão geral das fases do processo de desenvolvimento de software de XPlus



Fonte: Guimarães e Chaves (2009).

Na primeira fase, preparação, a equipe de desenvolvimento dá início ao diálogo com o cliente na tentativa de conhecer seu perfil e suas necessidades. E também é realizado o preparo de todo o ambiente de desenvolvimento, escolhendo as ferramentas, linguagem de programação e banco de dados, quando necessário. Então, na fase seguinte, é realizado o levantamento de requisitos do projeto para definir quais funcionalidades e interfaces serão implementadas, neste projeto optou-se por usar diagramas de casos de uso ao invés dos cartões de estórias e cartões de tarefas que o XPlus sugere.

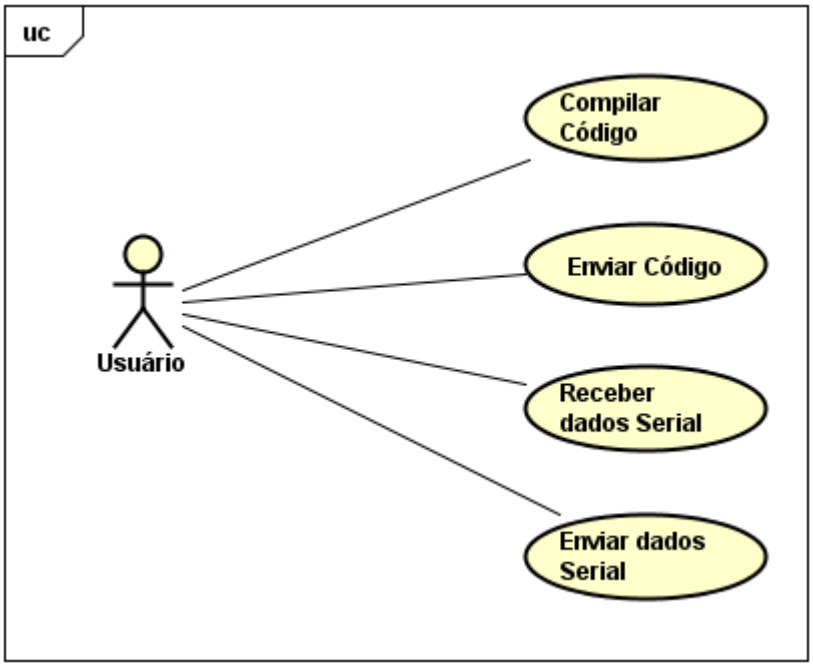
Na fase de implementação, o programador desenvolve o sistema de acordo com as especificações passadas pelo usuário que são representadas no diagrama de caso de uso e descrevem as funcionalidades da aplicação. Entretanto, a sua implementação requer uma série de passos que podem, em geral, serem executados em paralelo. Desse modo, para facilitar o desenvolvimento em paralelo pela equipe, o XPlus recomenda que cada funcionalidade seja quebrada em tarefas e desenvolva o código em pares. Para a implementação das tarefas que envolvem implementação de interface, a dupla deverá ser formada por um profissional de implementação e um profissional de interface que, em XP, é chamado de Designer de Interação.

Seguindo o ciclo, temos a fase de testes que consiste em validar o desenvolvimento do sistema de acordo com as heurísticas definidas pela equipe. A validação do código deve ocorrer de forma semelhante ao processo de XP, com Desenvolvimento Orientado a Testes e a validação da interface deverá ser feito pelo Designer de Interação. No final de cada iteração acontece o teste de aceitação onde o cliente deve verificar se o que foi implementado está de acordo com o que foi especificado.

4.3 Etapas de Desenvolvimento

Inicialmente foram realizadas reuniões com as equipes de desenvolvimento e equipe de hardware em conjunto com os coordenadores do projeto para compreensão dos requisitos necessários ao sistema. A partir destas reuniões foi definido o protótipo da interface gráfica da página Web que foi desenvolvida por outros membros do LARA e as funcionalidades do sistema, que são mostradas no diagrama de caso de uso da Figura 5.

Figura 5- Modelo de Caso de Uso do servidor de laboratório do projeto LARA.



Fonte: O autor.

O sistema deve permitir que o usuário compile o código Arduino sem interferir na conexão serial e retornar as mensagens de erro e dados gerais da compilação.

Ao enviar código, o sistema irá interromper a conexão serial temporariamente, para que seja possível transmitir o novo código ao dispositivo. Então o programa irá compilar e gravar o código no dispositivo, caso não contenha erros e, irá retornar os dados da compilação e envio de código ao usuário.

A função Receber dados Serial permite que o usuário obtenha os dados enviados pelo Arduino, assim como, a função Enviar dados Serial permite que o usuário escreva na conexão serial com o Arduino.

Para o desenvolvimento do protótipo do projeto foi escolhido o modelo de arquitetura cliente-servidor multinível. Esta arquitetura permite que a aplicação se comporte como cliente e, ao mesmo tempo, servidor. Em outras palavras, enquanto a aplicação funciona como servidor serial e de compilação, também pode funcionar como cliente para obter o código armazenado em algum servidor web da rede.

O XPlus sugere que seja criado modelos de interface gráfica do usuário (Guimarães e Chaves, 2009). No entanto, o gerenciamento do módulo servidor se dá apenas por comunicação via conexão de rede e não fornece acesso por uma interface gráfica, portanto a fase de prototipação não pôde ser contemplada no desenvolvimento

deste trabalho. Entretanto, houve consenso entre os membros do projeto LARA no momento de definir as funcionalidades que seriam implementadas para que o módulo cliente pudesse gerenciar o módulo servidor de acordo com suas necessidades e para criar uma interface gráfica de controle para o usuário. Desse modo, a interface inicial permitia que o usuário requisitasse ao servidor a compilação e envio do código remoto ao Arduino e a troca de informações pela interface de comunicação serial.

4.4 Implementação do SLL

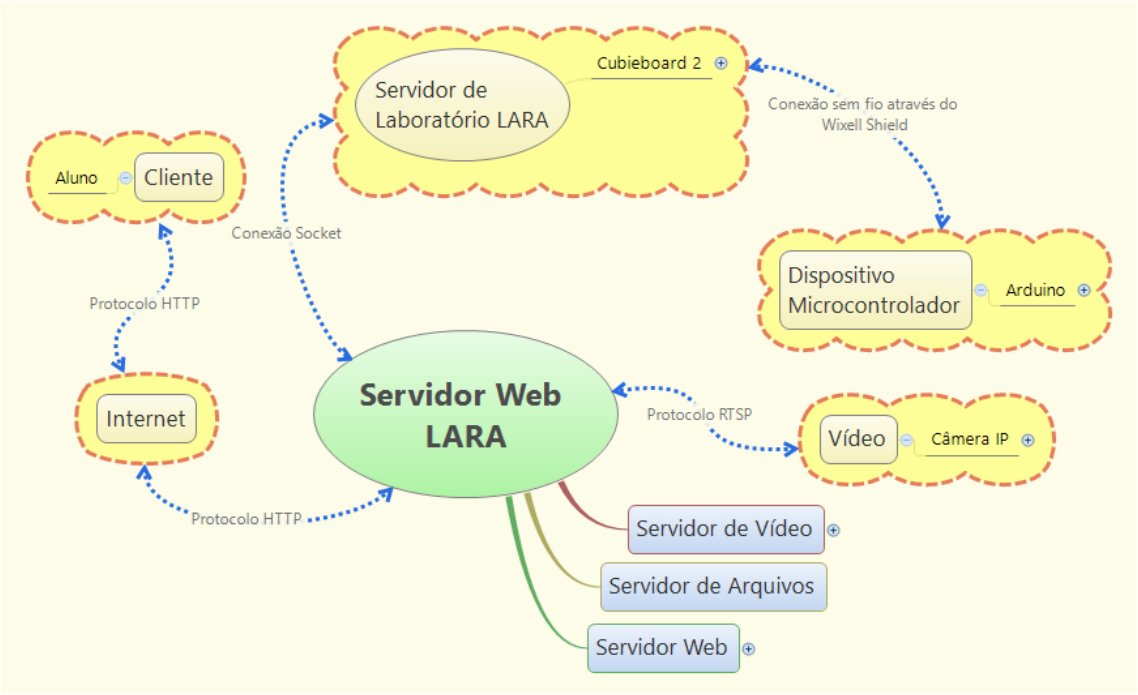
Nesta seção está descrito a arquitetura e o processo de implementação do código fonte do módulo servidor que faz a conexão entre o Servidor Web LARA e o Arduino.

Utilizando a abordagem da arquitetura cliente-servidor, o software foi desenvolvido com um módulo servidor que disponibiliza o acesso a recursos do sistema e o módulo cliente que acessa os recursos do servidor e faz a interação com o usuário do sistema foi desenvolvido por outra equipe do projeto LARA.

Como o módulo servidor apenas disponibiliza o acesso à recursos do sistema através de mensagens trocadas por conexão Socket, todo o processamento de informações e funcionamento interno fica transparente ao módulo cliente que recebe somente os resultados das requisições efetuadas. E o módulo cliente, por sua vez, tem o poder de requisitar acesso à recursos ao módulo servidor, realizar o gerenciamento de usuários e implementar uma interface de controle amigável para o sistema. O diagrama da Figura 6 ilustra a arquitetura planejada ao projeto LARA.

Neste diagrama temos que o Servidor LARA está diretamente conectado com o Servidor de Laboratório via conexão Socket para gerenciamento dos códigos que são enviados e para estabelecer a comunicação serial com o Arduino. O Servidor de Laboratório se comunica com o dispositivo microcontrolador utilizando o Wixel Shield. O Servidor Web LARA também está conectado com a câmera IP para exibição do experimento via protocolo RTSP, onde o streaming de vídeo é processado e enviado ao usuário do sistema pela página web. Na sua conexão com a Internet, o Servidor Web LARA fornece os serviços do laboratório por protocolo HTTP numa página web para o Cliente, que é usuário do sistema.

Figura 6 - Arquitetura do projeto LARA



Fonte: O autor.

O foco desse trabalho é o desenvolvimento do Servidor de Laboratório LARA que é composto pelo hardware e software. Na escolha do hardware foi considerado as características do LARA em utilizar tecnologias de código aberto e baixo custo, ao mesmo tempo em que o poder de processamento seja condizente com as necessidades do projeto. Portanto, foi escolhido o minicomputador Cubieboard 2[7] que é fruto de um projeto de hardware livre de baixo custo com processador ARMv7 dual core de 1GHz e 1GB de memória RAM, executando o sistema operacional Cubian R1, que é uma versão adaptada do Debian. No sistema operacional estão instalados o Arduino IDE, máquina virtual Java e o SLL, como mostra o diagrama na Figura 7.

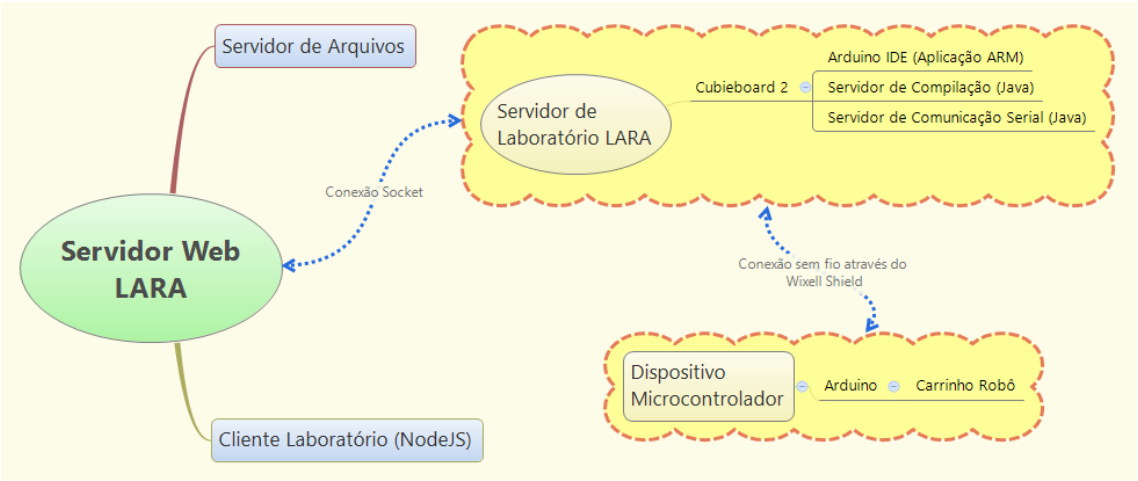
Embora a metodologia XPlus recomende que a fase de implementação seja executada em pares, o desenvolvimento do protótipo do SLL aconteceu individualmente por ser fruto do trabalho de conclusão de curso da UESB e, seguindo as regras da instituição, não pode ser elaborado em conjunto com outros discentes.

Entretanto, a implementação aconteceu com bastante interação entre os desenvolvedores de outras áreas no projeto LARA, permitindo que as tarefas de implementação fossem executadas ao mesmo tempo entre as equipes do projeto. As interações entre as equipes do projeto foram necessárias também para criar o protocolo

[7] http://cubieboard.org (Acessado em 19/06/2015)

de comunicação entre o SLL e módulo cliente, para que fosse capaz de atender às funcionalidades necessárias ao sistema de laboratório remoto.

Figura 7 - Modelo de conexão entre o Servidor de Laboratório e o Servidor Web e o Dispositivo Microcontrolador



Fonte: O autor.

Desse modo, o SLL ficou responsável por gerenciar a comunicação serial com o dispositivo Arduino, realizar os procedimentos para compilação e envio de novos códigos, tratar as novas conexões dos clientes e atender às suas solicitações durante a conexão. E o módulo cliente faz o intermédio com interface gráfica para o usuário ter acesso aos recursos do sistema.

Para a implementação do SLL, procurou-se uma linguagem de programação que fosse compatível com as mais diversas plataformas de computadores, inclusive com a Cubieboard 2, assim como, de código aberto, com ampla documentação e de fácil entendimento. Diante de tais necessidades, a linguagem Java acabou sendo empregada no desenvolvimento do sistema, por melhor atender aos requisitos citados. Dessa forma, surgiram os seguintes requisitos para a implementação do programa em Java no Cubian:

- Ambiente de desenvolvimento Java (JDK – Java Development Kit) de versão igual ou superior à 1.7
- A utilização da API para comunicação serial em Java: JavaComm. Esta API possui bibliotecas para estabelecer a comunicação via interface serial com o Arduino;
- Conexões via Socket (TCP) para estabelecer a conexão via rede de computadores entre o SLL e o módulo cliente e, assim, prover a troca de informações;

- Um arquivo de configuração contendo informações sobre as configurações do sistema em que o SLL está instalado, como por exemplo, a descrição da porta serial em que o Arduino se encontra conectado, o número da porta de comunicação via Socket em que o servidor irá aguardar novos clientes, entre outras configurações para o correto funcionamento do sistema;
- Dois arquivos *shell script* do sistema operacional contendo comandos pré-definidos para auxiliar na execução do ambiente de desenvolvimento Arduino, que irá compilar e enviar novos códigos ao microcontrolador;
- Uma conexão de rede estável entre o SLL e o módulo cliente, de modo que seja possível obter o mínimo atraso na troca de informações para uma melhor experiência de uso;
- O computador deve estar diretamente conectado ao Arduino, seja por um cabo USB, pelo Wixel Shield ou por qualquer outra tecnologia baseada na interface de comunicação serial;
- Ambiente de desenvolvimento Arduino na versão 1.5.6 BETA;
- E permissão do sistema operacional para acesso às portas de comunicação serial.

A utilização do SLL foi projetada para ser através da troca de mensagens com o módulo cliente via conexão de rede. Estas mensagens seguem um protocolo próprio desenvolvido especificamente para executar os comandos necessários ao sistema criado. O protocolo de comunicação segue o modelo de *comando+mensagem*, onde *comando* define a ação que o SLL irá executar e *mensagem* contém informações complementares para alguns comandos que o necessitem, como por exemplo, ao compilar ou enviar o código, *mensagem* irá conter o endereço web do código para baixá-lo no SLL. Quanto aos *comandos*, estes podem ser, resumidamente, para enviar a *mensagem* para a interface serial do Arduino; e compilar e verificar o código enviado pelo cliente.

Ao compilar ou enviar um código ao microcontrolador, o SLL armazena permanentemente o código baixado em uma pasta do sistema de arquivos junto com outros dois arquivos, onde um contém os dados da saída de erros e o outro os dados da saída padrão do compilador ao avaliar o código.

Durante o processo inicial de desenvolvimento do protótipo do SLL, foi definido que os dados recebidos pela interface de comunicação serial seriam armazenados num

buffer até que o módulo cliente solicitasse estes dados. A comunicação ocorria desta maneira pelo fato das primeiras versões implementadas no módulo cliente não terem sido capazes de manter uma conexão de rede aberta, o que gerava inúmeras solicitações ao SLL e, consequentemente, ocorria um alto atraso para o envio e recebimento de dados na comunicação serial. Para solucionar isto, o módulo cliente passou a utilizar ferramentas que mantém a conexão via rede aberta integralmente e, então, foi implementado no SLL o envio imediato (sem buffer) dos dados recebidos pela interface de comunicação serial. Este modo de comunicação eliminou o alto atraso para o envio e recebimento dos dados da comunicação serial.

4.5 Testes

Maldonado (2004) relata que "o processo de desenvolvimento de software envolve uma série de atividades nas quais, apesar das técnicas, métodos e ferramentas empregados, erros no produto ainda podem ocorrer". Visando minimizar ou, até mesmo, extinguir estes erros, técnicas para testes de software são aplicadas por uma análise dinâmica do produto gerado.

A realização dos testes do SLL seguiu o modelo do XPlus em aplicar o teste de unidade e teste de integração, porém foi descartado o teste de usabilidade devido ao software desenvolvido não possuir interface gráfica do usuário. O teste de unidade tem a capacidade de testar os menores recursos do sistema, geralmente sendo aplicado pelo desenvolvedor, cada método do sistema é executado individualmente utilizando como parâmetros valores que estão dentro do conjunto de dados do método. E o teste de integração valida o software testando a combinação dos módulos como um todo.

Desse modo, para avaliar o funcionamento do software protótipo foram realizados estes testes no código a fim de encontrar erros que podem passar despercebidos durante o processo de desenvolvimento.

4.5.1 Testes de unidade

Primeiramente foram realizados os testes de unidade usando a técnica de teste de caixa-preta e seguindo as funcionalidades descritas no diagrama de casos de uso da

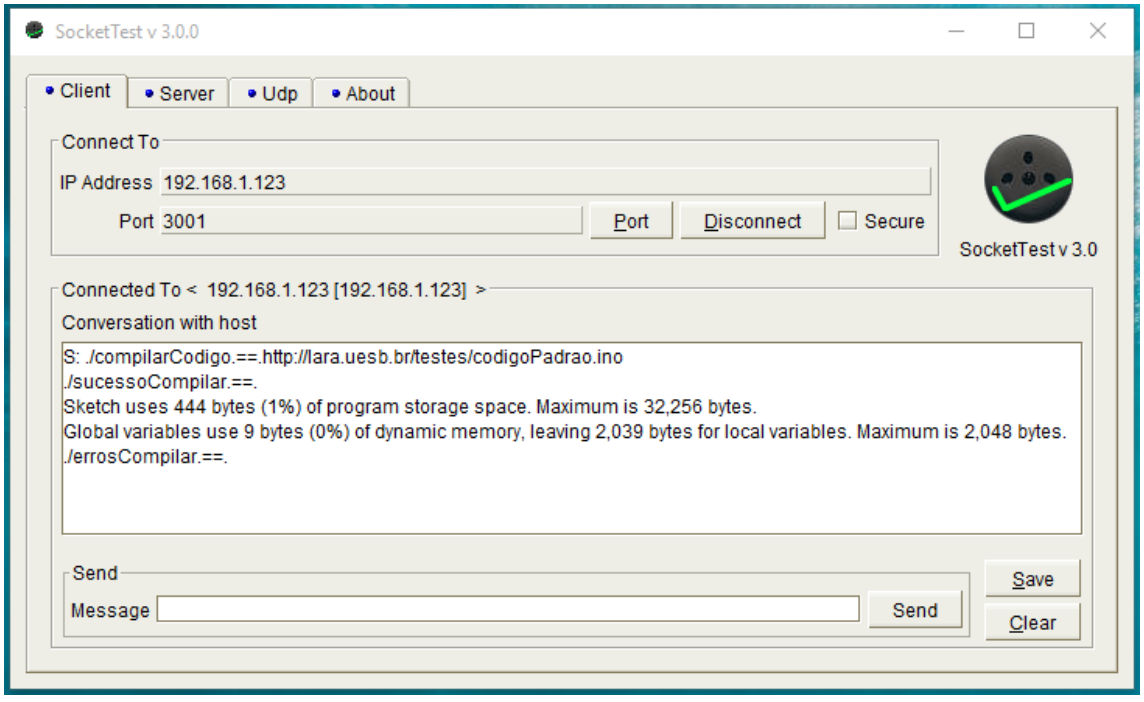
figura 5. Para a aplicação dos testes foi utilizada a ferramenta de testes SocketTest[8], um software disponibilizado na Internet, de código aberto, que possui suporte para realizar testes em conexões via Socket por interface gráfica. O funcionamento desta ferramenta, se dá apenas informando o endereço IP do servidor, a porta de conexão remota e a mensagem a ser enviada.

O primeiro caso de teste consistiu em validar o processo de compilação de código. Para isso, foram utilizados dois códigos Arduino, sendo o primeiro um código padrão Arduino, sintaticamente correto, e o segundo contendo um erro sintático. Estes códigos foram, então, armazenados num servidor web. Dessa forma, ao realizar o teste, é esperado que o SLL retorne a mensagem do compilador informando que o primeiro código está correto e que o segundo código não foi compilado por conter um determinado erro sintático.

Na Figura 8, temos o resultado do teste com o código Arduino que está armazenado em http://lara.uesb.br/testes/codigoPadrao.ino sem erros sintáticos. Foi passada a seguinte mensagem ao SLL:

"*./compilarCodigo.==.http://lara.uesb.br/testes/codigoPadrao.ino*"

Figura 8 - Resultado da compilação com código Arduino sem erros sintáticos



Fonte: O autor.

[8] Disponível em http://sockettest.sourceforge.net/ (Acessado em 15/05/2015)

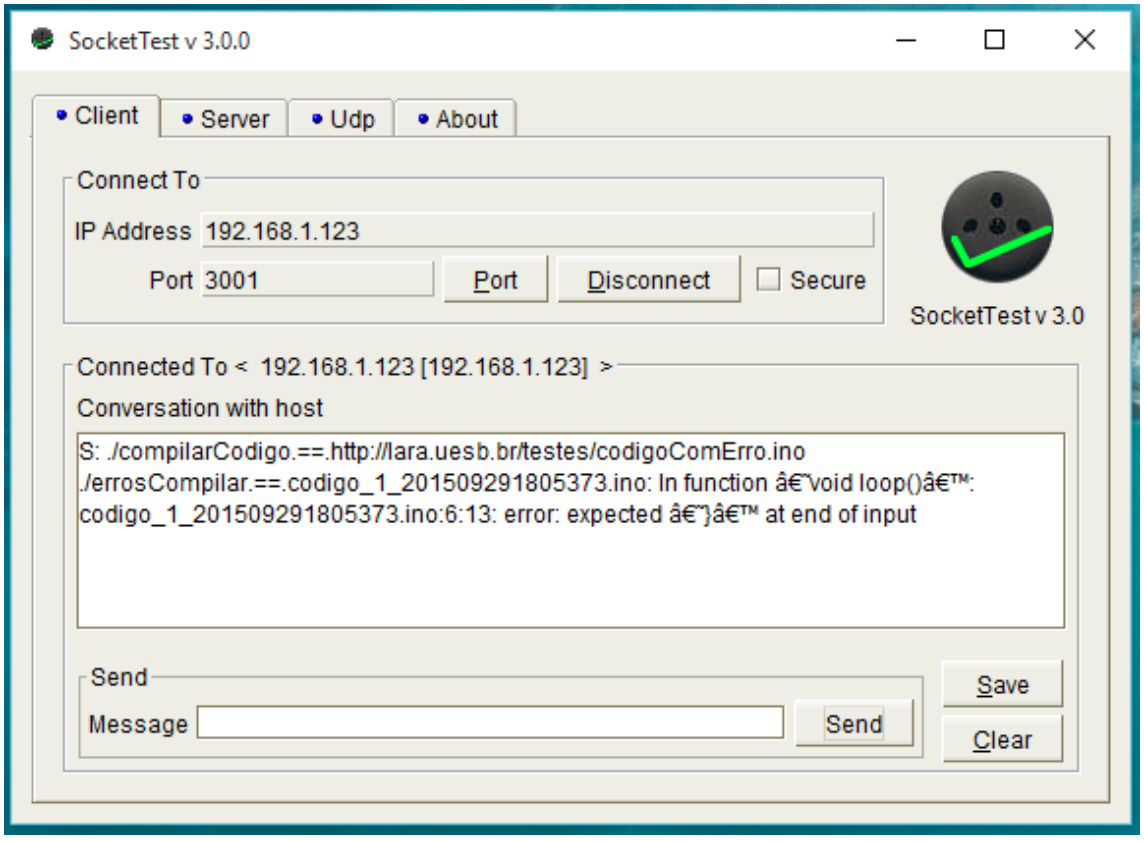
E é esperado como resposta o comando "*./sucessoCompilar.==.*" acrescido das informações de compilação do Arduino IDE e o comando "*./errosCompilar.==.*" sem acréscimo de informações.

Na Figura 9 é possível observar a mensagem de erro gerada ao compilar o código Arduino armazenado em http://lara.uesb.br/testes/codigoComErro.ino com erro sintático. A mensagem enviada neste processo teste foi:

"*./compilarCodigo.==.http://lara.uesb.br/testes/codigoComErro.ino*"

E é esperado como resposta as informações dos erros de compilação após a *flag* "*./errosCompilar.==.*".

Figura 9 - Resultado da compilação com código Arduino contendo erro sintático



Fonte: O autor.

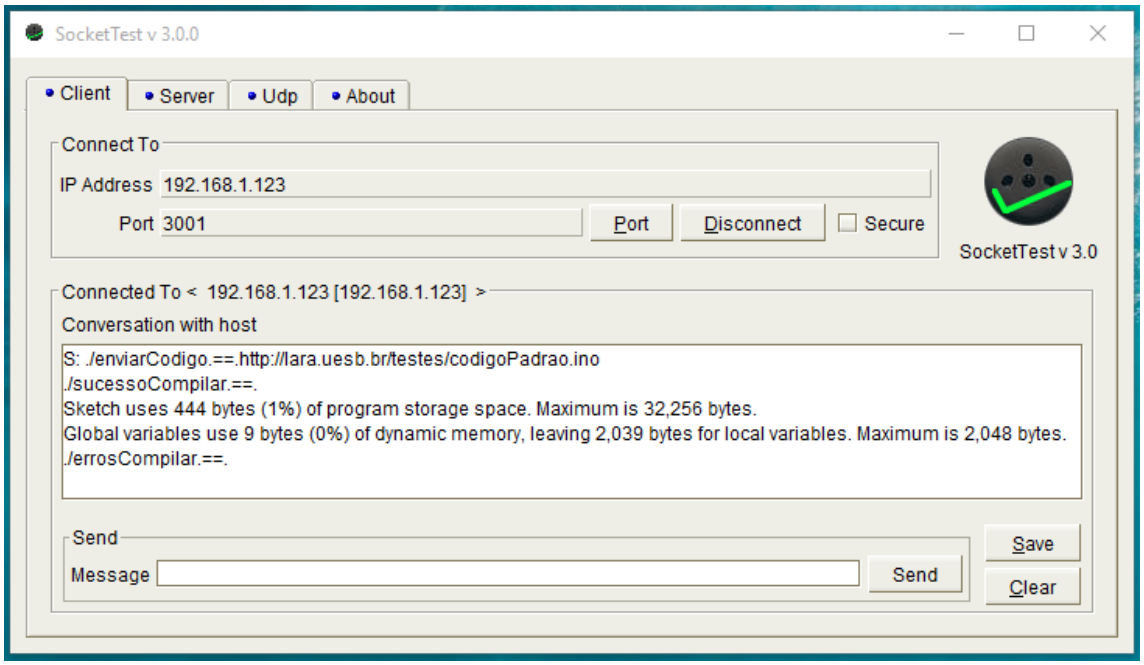
O segundo caso de teste de unidade consistiu em validar o processo de envio de código para o Arduino. Para isso, foram utilizados os mesmos códigos Arduino aplicados no primeiro teste. No teste é esperado que o SLL retorne a mensagem do compilador informando que o primeiro código está correto e que o segundo código não foi enviado por conter um determinado erro sintático.

Na Figura 10, temos o resultado do teste de envio de código Arduino que está armazenado em http://lara.uesb.br/testes/codigoPadrao.ino sem erros sintáticos. Foi passada a seguinte mensagem ao SLL:

"*./enviarCodigo.==.http://lara.uesb.br/testes/codigoPadrao.ino*"

E é esperado como resposta o comando "*./sucessoCompilar.==.*" acrescido das informações de compilação do Arduino IDE e o comando "*./errosCompilar.==.*" sem acréscimo de informações.

Figura 10- Resultado do envio de código Arduino sem erros sintáticos



Fonte: O autor.

Na Figura 11 é possível observar a mensagem de erro gerada ao enviar o código Arduino armazenado em http://lara.uesb.br/testes/codigoComErro.ino com erro sintático. A mensagem enviada neste processo teste foi:

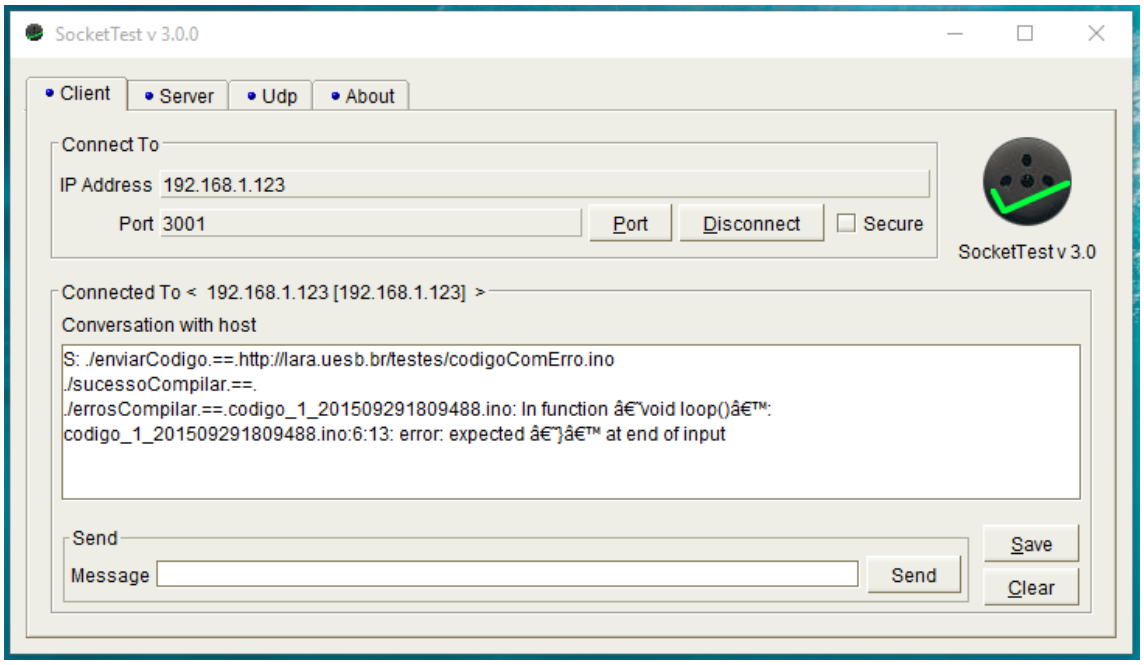
"*./compilarCodigo.==.http://lara.uesb.br/testes/codigoComErro.ino*"

E é esperado como resposta as informações dos erros de compilação após a *flag* "*./errosCompilar.==.*".

O terceiro teste verificou o recebimento de dados da interface de comunicação serial do Arduino. Para isso, foi escrito um código Arduino que escreve os números de 0 a 9, cada um em uma linha, e enviado ao dispositivo. Portanto, é esperado pelo SocketTest o recebimento da *flag* "./serial.==." acrescida da sequência numérica. A

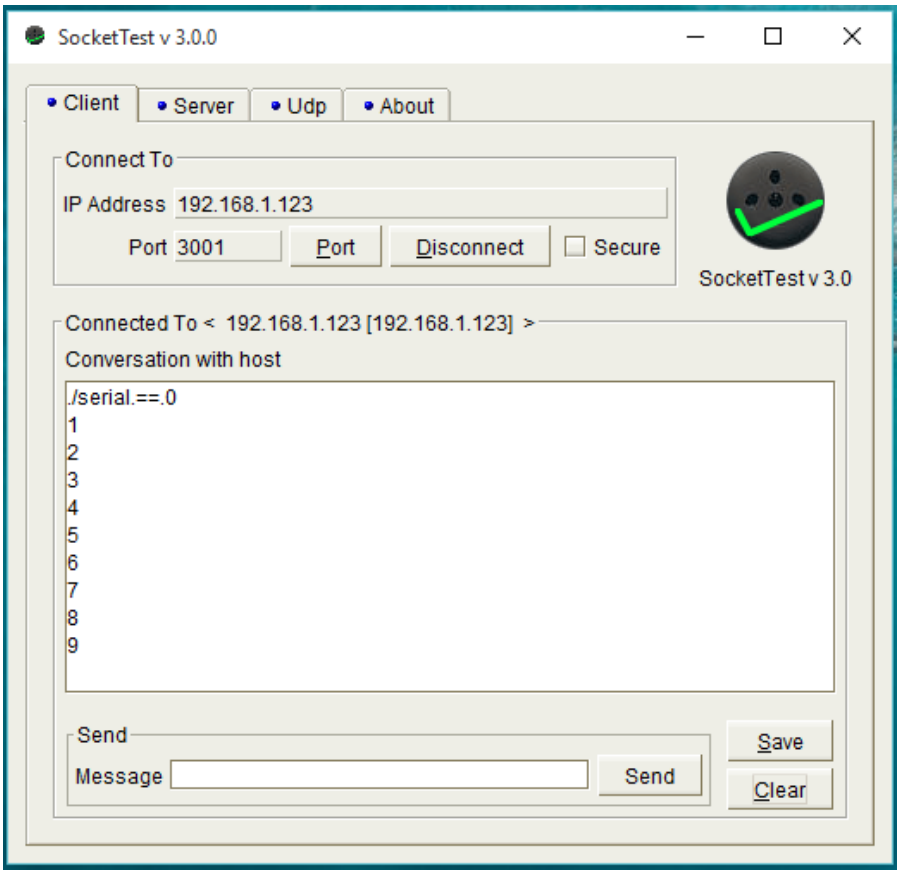
Figura 12 ilustra o recebimento dos dados da interface de comunicação serial do Arduino.

Figura 11 - Resultado do envio de código Arduino contendo erro sintático



Fonte: O autor.

Figura 12 – Recebimento de dados da interface serial do Arduino

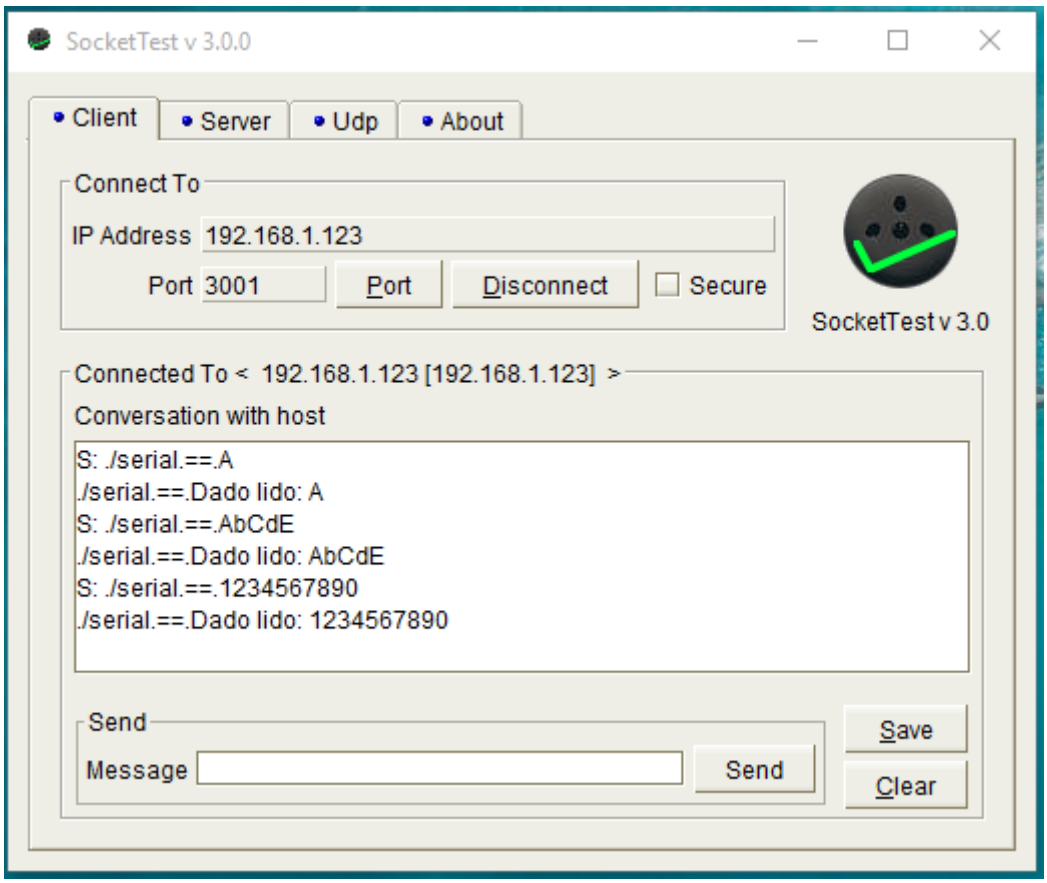


Fonte: O autor.

O próximo teste consistiu em verificar o correto envio de dados pela interface de comunicação serial. Então, foi escrito um código Arduino que sempre retorna “Dado lido:” concatenado com a mensagem enviada. Desse modo, ao enviar a *flag “./serial.==.”* concatenado com “A”, é esperado a resposta “./serial.==.Dado lido: A”, o mesmo vale para “AbCdE” e “1234567890”.

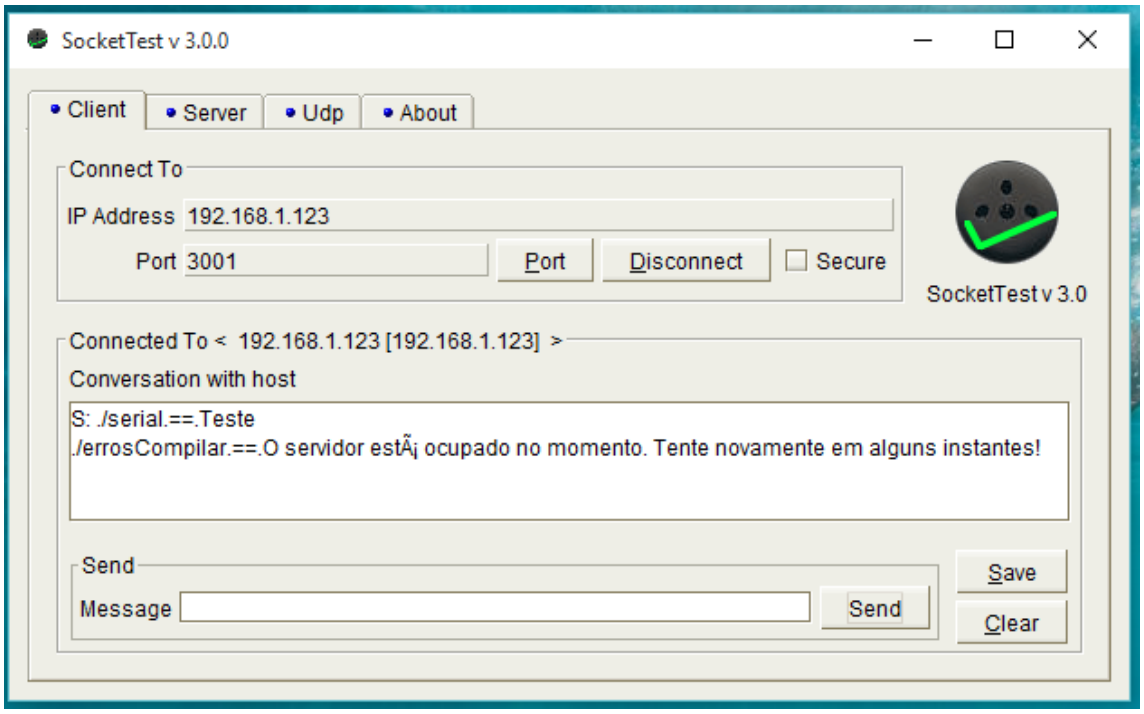
O último teste aplicado consistiu em verificar o comportamento do SLL ao receber o comando para enviar dados à serial enquanto um código está sendo enviado para o Arduino. Para realizar este teste, foi necessário utilizar duas instâncias do SocketTest, uma tentou enviar dados à interface de comunicação serial do Arduino ao mesmo tempo que a outra instância tinha dado o comando para enviar um novo código ao dispositivo. Logo, é esperado que o SLL informe à instância do SocketTest que enviou dados à serial que o sistema está ocupado no momento. A Figura 14 ilustra o resultado deste teste.

Figura 13 – Teste de envio de dados via comunicação serial



Fonte: O autor.

Figura 14 – Teste de envio de dados pela comunicação serial ao mesmo tempo que um código está sendo enviado ao Arduino



Fonte: O autor.

4.5.2 Teste de integração

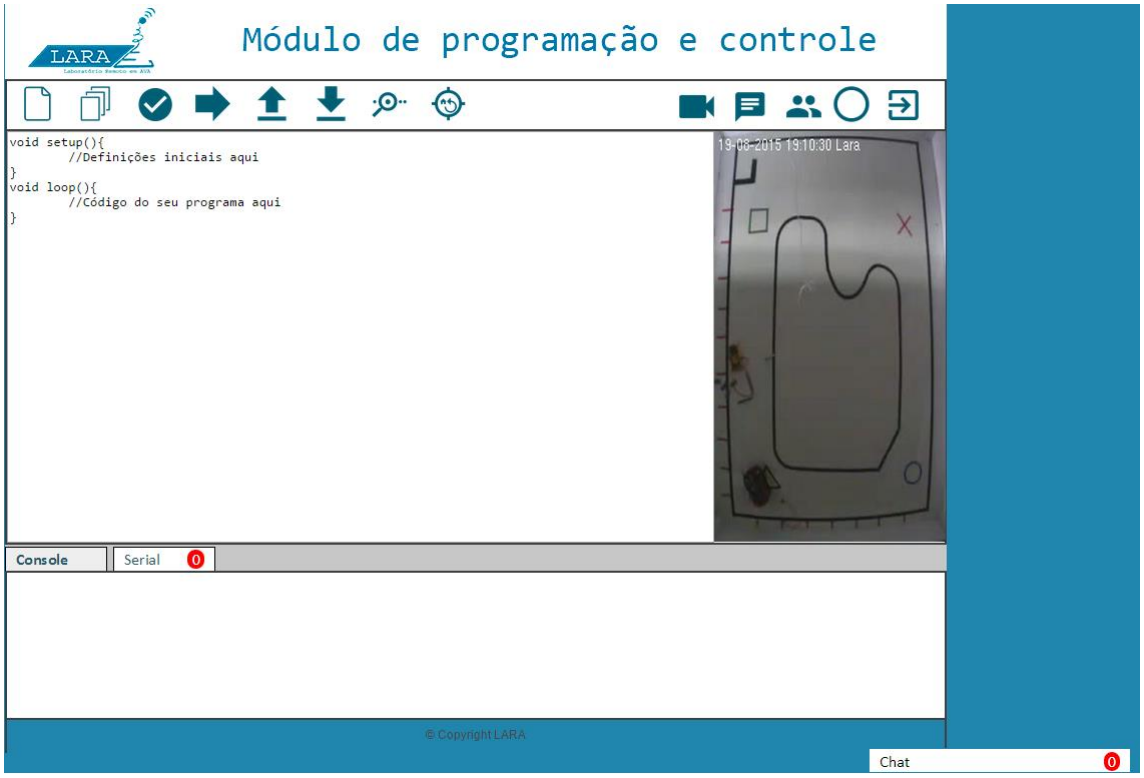
Teste de integração é a fase do teste de software em que módulos são combinados e testados em grupo. Este é um teste que ocorre após o teste de unidade e seu objetivo é verificar se os recursos implementados no SLL, irão funcionar satisfatoriamente no módulo cliente. Dessa forma, a realização deste teste aconteceu em conjunto com o módulo cliente, Módulo de Programação e Controle, desenvolvido por outros membros do projeto LARA, que possui interface gráfica acessível ao usuário do sistema por um navegador web, como é mostrado na Figura 15.

Em um primeiro momento durante os testes, a compilação e envio de código Arduino funcionaram dentro do esperado, mas a interface de comunicação serial apresentou um alto tempo de atraso no envio e recebimento de dados, chegando até 30 segundos de atraso.

Depois de algumas pesquisas, foi observado que o atraso ocorria pela tecnologia usada na implementação da página web, que sempre criava uma nova conexão socket para solicitar os dados armazenados no buffer da comunicação serial do SLL. Uma solução encontrada foi alterar a tecnologia do módulo cliente para uma que tivesse

suporte de manter a conexão socket aberta integralmente. Com essa alteração, deixou de ser necessário a utilização do buffer para comunicação serial, passando a receber os dados da serial automaticamente, na medida que o SLL recebe do Arduino e com isso, o tempo de atraso passou para menos de 1 segundo.

Figura 15 - Módulo de Programação e Controle - LARA



Fonte: O autor.

4.6 Conclusão

Com a necessidade de se desenvolver um sistema de laboratório remoto para o projeto LARA, que tem como requisitos recursos inovadores no controle de dispositivos robóticos, criou um novo campo de estudo para implementação de um servidor de laboratório que faz o intermédio na comunicação entre o dispositivo Arduino e o Servidor Web LARA, uma vez que, publicações acadêmicas não focam no processo de desenvolvimento do software servidor de laboratório, muitas vezes utilizando softwares prontos e pagos.

Então, foi necessário desenvolver o SLL seguindo uma adaptação da metodologia de desenvolvimento XPlus com o diagrama de caso de uso que descreve as funcionalidades implementadas.

Para verificar o correto funcionamento do sistema implementado foram aplicados testes que avaliaram o comportamento do sistema para determinadas situações. Estes testes foram de grande importância para perceber falhas de implementação e integração com o cliente (Servidor Web LARA) que não eram visíveis no código fonte.

Os testes comprovaram que o sistema implementado foi bem analisado na fase inicial do sistema, sendo observado um baixo índice de erros. Mostrando que os requisitos elucidados estão bem estruturados e foram implementados corretamente para o LARA. Permitindo o desenvolvimento e funcionamento de um ambiente de estudo de linguagem de programação e robótica, utilizando o Arduino, através de um laboratório remoto disponível na Internet.

Como não houve saídas não previstas nos testes isto requer uma aplicação de novos casos de testes.

5 Conclusões

Neste trabalho foi abordado o tema de laboratórios remotos, que tem sido de grande interesse pela comunidade científica nos últimos anos. Tendo como objetivo geral desenvolver um laboratório remoto de robótica integrado ao ambiente virtual de aprendizagem no Moodle, baseado em tecnologias de código aberto para ensinar linguagem de programação, algoritmos, inteligência artificial e robótica educativa.

Como visto, a elaboração deste trabalho veio da necessidade de se resolver o problema de comunicação entre o Servidor Web e um robô autodirigido tendo o Arduino como dispositivo microcontrolador, proposto no projeto de pesquisa LARA. Suas características de laboratório remoto permitem flexibilidade de acesso em questão de local e horário. Entretanto, grande parte dos sistemas de ensino através da Internet utilizam tecnologias para transmissão de informação e não necessariamente conhecimento. Nesse sentido, esse projeto buscou construir uma interface de controle de modo que possa ser atingido o objetivo geral do projeto que é desenvolver um ambiente que favoreça a construção do conhecimento fundamentado em toda uma estrutura pedagógica capaz de fazer uso das ferramentas de controle disponíveis ao usuário.

Nesse sentido, foram abordadas as definições da plataforma Arduino e de alguns dos processos de comunicação entre dispositivos eletrônicos. Bem como, a apresentação de alguns trabalhos relacionados com o desenvolvimento de laboratórios remotos.

Com o trabalho foi possível desenvolver o SLL, um software que traz como principal funcionalidade promover a comunicação entre o dispositivo físico de experimento e o cliente (Servidor Web LARA) que fornece a interface de acesso ao usuário do sistema, além de permitir que o usuário possa compilar e enviar o código Arduino remotamente.

Para análise do sistema foram realizados testes software de unidade e de integração. Os dados gerados foram analisados de forma quantitativa, sendo verificado se para cada entrada, foi gerado a saída desejada. Durante os testes de unidade não foram encontrados erros, o que indica que novos testes devem ser feitos.

Os testes de integração possibilitaram agregar o SLL à interface do Módulo de Controle e Programação. Durante o mesmo foi possível verificar que um dos requisitos funcionais, o tempo de espera para a troca de dados da conexão serial, não foi atingido. Esse fato se deu por uso de uma tecnologia usada no desenvolvimento da interface web

que não era compatível com a implementação de conexão Socket feita no SLL, que tem um melhor desempenho quando o cliente mantém a conexão aberta continuamente. Mas durante o projeto, este problema foi solucionado e agora a comunicação serial ocorre com atraso inferior a 1 segundo.

O SLL é responsável por permitir a comunicação e gerenciamento do cliente ao dispositivo microcontrolador. Para isso, o SLL permite que o cliente possa enviar e receber dados da porta de comunicação serial e enviar e compilar código Arduino, retornando as informações do compilador.

Em meio às dificuldades encontradas durante o desenvolvimento do SLL, o resultado final se mostrou melhor que o esperado. Pois o SLL possibilitou a comunicação e gerenciamento entre uma aplicação web com um dispositivo microcontrolador através da rede de computadores e disponibilizado para acesso pela Internet, utilizando tecnologias de código aberto e equipamentos de baixo custo, além de colocar em prática conhecimentos adquiridos no decorrer da graduação em Ciência da Computação num projeto que visa semear o conhecimento com todos.

O SLL já está sendo utilizado no sistema de laboratório remoto do LARA por alunos ingressantes do curso de Ciência da Computação da UESB. E sua implementação permite que modificações possam ser realizadas para inserir novas funcionalidades e recursos ao sistema

5.1 Contribuições

O uso da tecnologia web se mostrou adequada para o desenvolvimento do SLL que possibilita ao usuário de um laboratório remoto controlar um dispositivo robótico. Com isso foi possível desenvolver um laboratório remoto como especificado no projeto LARA.

Outro fato importante que foi possível aplicar e aprimorar o conhecimento adquirido no decorrer da graduação para construção de um projeto de pesquisa, que busca levar o acesso e ensino de controle de dispositivos robóticos a todos.

5.2 Trabalhos futuros

Embora esteja de acordo com o especificado, com um bom desempenho, o SLL possui potencial para melhorias relevantes. Então, como trabalho futuro sugere-se:

1. A implementação da arquitetura orientada a recursos REST/ROA no processo de comunicação entre o SLL e o Servidor Web LARA,
2. Elaboração e aplicação de novos testes de software para detecção de erros;
3. Adicionar funcionalidades que possam surgir nos requisitos do projeto e nos resultados dos testes com os usuários do sistema.

Referências

ALBUQUERQUE, Hugo Rodrigues de. **Um estudo sobre pilhas Bluetooth e suas limitações em sistemas embarcados**. Universidade Federal de Pernambuco, 2010.

ÁLVARES, Alberto José; FERREIRA, João Carlos Espíndola. **Metodologia para implantação de laboratórios remotos via Internet na área de automação da manufatura**. In: 2o Congresso Brasileiro de Engenharia de Fabricação (COBEF), Uberlândia, MG. 2003.

CASTELLS, Manuel. **A Galáxia Internet: reflexões sobre a Internet, negócios ea sociedade**. Zahar, 2003.

CHELLA, Marco Túlio, FERREIRA, Elnatan Chagas. **Arquitetura para Desenvolvimento de Experimentos Remotos com Aplicações em EAD**. Revista Brasileira de Aprendizagem Aberta e a Distância, São Paulo, Dez. 2007.

CRUZ, Marcia Kniphoff da et al. **Controle de Kit de Robótica através de Laboratório Remoto pela Internet: uma Aplicação para a Formação Docente e para a Educação Básica**. In: Anais do Simpósio Brasileiro de Informática na Educação. 2009.

D'ABREU, J.V.V. ; MARTINS, M. C. . **Robótica Educacional e Acesso Remoto: relato de uma experiência**. In: Iº Simpósio Internacional sobre Novas Competências em Tecnologias Digitais Interativas na Educação, 2007, Campinas. Iº Simpósio Internacional sobre Novas Competências em Tecnologias Digitais Interativas na Educação, 2007.

FERNANDES, Amadeu Socorro Lopes. **Comunicação Ad Hoc em Equipas de Robôs Móveis Utilizando a Tecnologia ZigBee**. Universityof Coimbra, 2012.

FUCK, Marcos Paulo; VILHA, Anapatrícia Morales. **Inovação Tecnológica: da definição à ação**. Contemporâneos: Revista de Artes e Humanidades, p. 1-21, 2011.

GUIMARÃES, Carina Piauhy; CHAVES, Christina von Flach Garcia. **XPlus: Integrando o Design de Interfaces Centrado na Experiência do Usuário ao Processo de Desenvolvimento de Software com eXtreme Programming**.UFBA. Salvador, 2009.

KOBAYASHI, Carlos Yassunory. **A Tecnologia Bluetooth e aplicações**. USP. São Paulo, 2004.

MALDONADO, José Carlos et al. **Introduçao ao teste de software**. São Carlos, 2004.

MARTINS, Janae Gonçalves et al. **Aprendizagem baseada em problemas aplicada a ambiente virtual de aprendizagem**. 2002. Tese de Doutorado. Universidade Federal de Santa Catarina, Centro Tecnológico. Programa de Pós-Graduação em Engenharia de Produção.

NUMAJIRI, D. A. **Desenvolvimento de um sistema de controle de dispositivos via acesso remoto**. Lavras: UFLA, 2003.

PRETTO, Nelson De Luca; CORDEIRO, Salete Noro; DOS SANTOS OLIVEIRA, Washington. **Produção Cultural E Compartilhamento De Saberes Em Rede: Entraves E Possibilidades Para A Cultura Ea Educação**. Educação em Revista, v. 29, n. 3, p. 17-40, 2013.

SABBATINI, Renato ME**. Ambiente de Ensino e Aprendizagem via Internet A Plataforma Moodle**. São Paulo: Instituto EduMed, 2007.

SILVA, Alzira Ferreira da. Lopes, Maísa Soares dos Santos. Trindade, Roque Mendes Prado. **Laboratório Remoto em AVA**. Projeto de Pesquisa. Departamento de Ciências Exatas e Tecnológica. Universidade Estadual do Sudoeste da Bahia – UESB. Vitória da Conquista – BA, 2014.

TANENBAUM, Andrew S. **Redes de computadores**. Pearson Educación, 2003.

VILAÇA, Márcio Luiz Corrêa. **Pesquisa e ensino: considerações e reflexões**. Revista e-scrita: Revista do Curso de Letras da UNIABEU, v. 1, n. 2, p. 59-74, 2010.

ZEPELIN, S. R.; BORDA, R. R. de S. **Projeto de validação experimental de topologias de robôs móveis aplicadas à robótica educacional**. XL Congresso Brasileiro de Educação em Engenharia, 2012.

Apêndice A

Código do *shell script* para envio de código Arduino

```
#!/bin/bash
diretorio="/home/cubie/codigos"
arduino --port $1 --upload $diretorio/$2/$2.ino >
$diretorio/$2/$3 2> $diretorio/$2/$4
```

Apêndice B

Código do *shell script* para compilação de código Arduino

```
#!/bin/bash
diretorio="/home/cubie/codigos"

arduino --verify $diretorio/$1/$1.ino > $diretorio/$1/$2 2>
$diretorio/$1/$3
```