Fundação Educacional do Município de Assis
Instituto Municipal de Ensino Superior de Assis
Campus "José Santilli Sobrinho"

NILTON URIAS DA CRUZ JUNIOR

INTEGRAÇÃO DA PLATAFORMA ARDUINO NA WEB APOIADA POR CHROME APPS E CHROME EXTENSIONS

Assis

2016

Fundação Educacional do Município de Assis
Instituto Municipal de Ensino Superior de Assis
Campus "José Santilli Sobrinho"

NILTON URIAS DA CRUZ JUNIOR

INTEGRAÇÃO DA PLATAFORMA ARDUINO NA WEB APOIADA POR CHROME APPS E CHROME EXTENSIONS

Projeto de pesquisa apresentado ao curso de Bacharelado em Ciências da Computação do Instituto Municipal de Ensino Superior de Assis – IMESA e a Fundação Educacional do Município de Assis – FEMA, como requisito parcial à obtenção do Certificado de Conclusão.

Orientador: Prof. MSc. Guilherme de Cleva Farto

Área de Concentração: Informática

Assis

2016

FICHA CATALOGRÁFICA

JUNIOR, Nilton Urias da Cruz.

INTEGRAÇÃO DE PLATAFORMA ARDUINO NA WEB APOIADA POR CHROME APPS E CHROME EXTENSIONS / Nilton Urias da Cruz Junior. Fundação Educacional do Município de Assis –FEMA – Assis, 2016.

43p.

Trabalho de Conclusão de Curso – Instituto Municipal de Ensino Superior de Assis - IMESA

Orientador: Prof. MSc. Guilherme de Cleva Farto

1. Arduino. 2. Chrome Apps. 3. Chrome Extensions. 4. Google Chrome. 5. Integração. 6. Comunicação serial.

CDD: 001.6

Biblioteca da FEMA

INTEGRAÇÃO DE PLATAFORMA ARDUINO NA WEB APOIADA POR CHROME APPS E CHROME EXTENSIONS

NILTON URIAS DA CRUZ JUNIOR

Trabalho de Conclusão de Curso apresentado ao Instituto Municipal de Ensino Superior de Assis, como requisito do Curso de Graduação, avaliado pela seguinte comissão examinadora:

Orientador: _____
Prof. MSc. Guilherme de Cleva Farto

Examinador: _____
Prof. Dr. Luiz Carlos Begosso

Assis/SP
2016

DEDICATÓRIA

Dedico este trabalho em especial a
DEUS e minha família, mãe, irmã, na
qual foram pessoas que me deram forças e
lutaram junto comigo para que este sonho
se tornasse realidade.

AGRADECIMENTOS

A todos os professores que estiveram comigo nessa caminhada e que contribuíram de alguma maneira para o meu crescimento pessoal e profissional.

À minha família, por estar sempre ao meu lado me apoiando e fortalecendo.

Aos amigos que cultivei durante o curso que me ajudaram nessa caminhada.

Ao meu orientador professor Guilherme de Cleva Farto, pelo apoio nas pesquisas e pelo conhecimento transmitido para que esse trabalho pudesse ser realizado.

Por fim agradeço a todos que contribuíram diretamente ou indiretamente para a minha formação e na execução deste trabalho, deixo aqui meu muito obrigado a todos.

"É ótimo celebrar o sucesso, mas mais importante ainda é assimilar as lições trazidas pelos erros que cometemos"

Bill Gates

RESUMO

A prototipação de projetos digitais e eletrônicos, utilizando plataformas que auxiliam o usuário, melhorando o aprendizagem e auxiliando na resolução de problemas de maneira mais eficiente e rápida. O objetivo deste trabalho é integrar a plataforma Arduino com o *Chrome Apps/Extensions*, permitindo que o usuário receba e envie informações pela porta *serial e*, dessa forma, manipule as informações geradas pelos sensores e atuadores conectados ao Arduino. A aplicação desenvolvida utilizando os conceitos de Chrome Apps, e juntamente com a API chrome.serial utilizado para receber e enviar dados para o Arduino foram realizados, onde, com esses conceitos utilizados pode-se analisar que a proposta do trabalho obteve resultado positivo, onde os componentes estavam funcionando.

Palavras-chave: Arduino, Chrome Apps, Chrome Extensions, integração, comunicação serial.

ABSTRACT

The prototyping of digital and electronic projects using platforms that help the user, improving learning and helping to solve problems more efficiently and quickly. The objective of this work is to integrate the Arduino platform with Chrome Apps / Extensions, allowing the user to receive and send information through the serial port and thus manipulate the information generated by the sensors and actuators connected to the Arduino. The application developed using the concepts of Chrome Apps, and together with the chrome.serial API used to receive and send data to the Arduino were performed, where, with these concepts used can be analyzed that the proposed work obtained positive results where the components were functioning.

Keywords: Arduino, Chrome Apps, Chrome Extensions, integration, serial communication.

LISTA DE ILUSTRAÇÕES

SUMÁRIO

1 – INTRODUÇÃO

Nos últimos anos, houve um grande aumento em ideias, na busca por soluções e ideias inovadoras que resultaram em projetos que podem ser adotados em distintas plataformas computacionais, como, por exemplo, o Arduino, Raspberry PI, Intel Galileo, entre outros. Dentro destes projetos, alguns se destacam, ajudando em estudos, desde a plataforma, como também nos componentes utilizados. Com diversos blogs, sites, fóruns milhares de projetos exibem projetos propostos e experimentais onde as pessoas realizam com os seus Arduinos, segundo McRoberts (2010).

Com o avanço da tecnologia, vários dispositivos foram criados, colaborando com as pessoas em várias áreas, com o auxílio da Internet. Com base nisso, surge o termo *Internet of Things* (IoT), que, de maneira resumida, é definida pela possibilidade de, conectar objetos à Internet. Junto aos conceitos de IoT, aumentou-se os números de projetos visando beneficiar novos tipos de dados. Segundo Pires et al., (2015), os objetos são interconectados a outros recursos, e podem ser controlados por meio da Internet, surgindo assim uma abundância de aplicações, beneficiando com novos tipos de dados.

O grande aumento de *smartphones* e *tables*, alavancou o número de dispositivos conectados à Internet em 2010 (EVANS, 2011), e com o crescimento de inúmeros dispositivos, e várias ideias e projetos são concebidos.

Alguns projetos chamam a atenção, como *Paramecium/Arduino Interface*, onde é possível controlar microorganismos por meio de sensores de ondas cerebrais de brinquedo, segundo GEVA (2013). Outro projeto, Luva-Sonar ajuda pessoas com deficiência visual, substituindo o cão-guia. A luva possui sensores ultrassônicos que emitem ondas sônicas inaudíveis, com as informações dos sensores, utiliza-se servo-motores para fazer uma pressão na costa da mão, conforme o objeto estiver mais perto, maior será a pressão. O kit BITalino contém sensores que captam algumas atividades do corpo humano, algumas atividades que ele verifica são, a *Electromyography* (EMG) onde contra a ativação muscular, *Electrodermal Activity* (EDA) que verifica os níveis de atividade da pele, *Electrocardiogram* (ECG) controla a frequência cardíaca, entre outros (LOMAS, 2013).

Com as dificuldades encontradas para a confecção de protótipos, a plataforma Arduino foi inicialmente concebida para suprir essas dificuldades, assim gerando um meio fácil e rápido de prototipação, pois não é necessário possuir conhecimentos técnicos de eletrônica e/ou de engenharia elétrica. Com o avanço da prototipagem, usuários auxiliam publicando soluções de problemas em sites, explicando como deve ser utilizado o componentes, onde os *datasheets* deles acaba sendo complicado de entender e analisar como é utilizado, assim não necessitando de conhecimento técnico (BRAZILEIRO, 2013).

No Arduino, há distintos componentes para a entrada e saída de dados, assim como em um computador tradicional, que são utilizados para a integração com o ambiente externo e com os usuários finais. Além dos sensores tradicionais, placas auxiliares chamadas de *Shield*, onde ele é acoplado facilmente no Arduino, assim facilitando, e agilizando a prototipação fornecendo recursos adicionais à plataforma Arduino, facilitando a prototipação de projetos digitais que necessitam de diferentes funcionalidades como, por exemplo, GPS, cartão de memória, sensores de temperatura e outros.

1.1 OBJETIVOS

O principal objetivo deste projeto é o de pesquisar os conceitos das plataformas Arduino, Chrome Apps, e Chrome Extensions, bem como as ferramentas necessárias para o desenvolvimento de uma aplicação Web que possibilite a integração de projetos Arduino com base no Chrome Apps e Chrome Extensions.

Dessa forma, espera-se tornar possível a manipulação e a interação de projetos desenvolvidos com Arduino por meio de uma interface baseada na Web. A aplicação proposta adotará os recursos da plataforma Chrome para se comunicar com o dispositivo Arduino por meio de classes que manipulam a porta Serial.

1.2 JUSTIFICATIVAS

Tendo em vista a comunicação com o Arduino e a Internet, para o envio e o recebimento de dados, um meio fácil e rápido para esta comunicação é a conexão direta do Arduino com a Internet, assim podendo recolher dados de um sensor e enviar para servidores. Pela facilidade do Arduino, é muito utilizado como meio de ensino de lógica de programação e também para o ensino de eletrônica, simplificando a prototipação e desenvolvimento de projetos eletrônicos.

1.3 MOTIVAÇÃO

O desenvolvimento deste projeto de pesquisa consiste na integração do Arduino com uma interface Web baseada em Chrome para fornecer novos meios de interação dos projetos eletrônicos.

Assim, utilizando o conceito estudado sobre a plataforma Chrome Apps e Chrome Extensions para o desenvolvimento de um ambiente, servindo para aprendizagem sem a necessidade de criar um ambiente para a utilização dos componentes do Arduino.

Outra motivação é a chance de contribuir com os projetos *open-source* facilitando no desenvolvimento de prototipações rápidas, recuperando e enviando valores na Web, sem a necessidade de um desenvolvimento específico.

1.4 ESTRUTURA DO TRABALHO

O trabalho está estruturado nas seguintes partes:

O presente trabalho está dividido em seis capítulos. O capítulo 1, apresenta a Introdução, os objetivos, justificativas e motivações para o desenvolvimento da pesquisa. O Capítulo 2 aborda a plataforma Arduino. O Capítulo 3 apresenta O Chrome Apps e Chrome Extensions. O Capítulo 4 aborta sobre a Proposta de Trabalho. O Capítulo 5 apresenta o Desenvolvimento de Trabalho. O Capítulo 6 apresenta a Conclusão.

2 – PLATAFORMA ARDUINO

O objetivo deste capítulo é apresentar os principais conceitos da plataforma *Arduino*, bem como as definições fundamentais e básicas para o entendimento de prototipação rápida.

Também serão explorados os principais componentes e *shields* que podem ser adotados em um projeto *Arduino* para facilitar a prototipação, por exemplo, de um projeto industrial.

2.1 HISTÓRICO

A plataforma *Arduino*, originou-se na Itália em 2005, com o objetivo de criar um meio de prototipação rápida e fácil, minimizando o conhecimento de componentes e na leitura de *datasheets*. Atualmente o *Arduino* utiliza o conceito do *open-source*, onde é livre para a contribuição da sociedade, assim encontra-se outros modelos do *Arduino* criado pela sociedade, desde modelos nacionais, como internacionais.

Segundo McRoberts (2010, p.22), “Em termos práticos, um *Arduino* é um pequeno computador que você pode programar para processar entradas e saídas entre o dispositivo e os componentes externos conectados a ele”. Ele utiliza um micro controlador com acesso de Entrada/Saída (I/O), onde é verificado entrada de dados como de sensores, e saída de dados como *LCD*. O micro controlador utilizado é o *Atmega*, onde é utilizado para controlar as ações utilizando de uma programação.

Atualmente, o *Arduino* é bastante utilizado no ambiente acadêmico, sendo fácil para a aprendizagem de vários componentes. Com a fácil programação, o *Arduino* possibilita modificações na programação reutilizando o próprio equipamento.

Em 2005, na Itália na cidade de Ivrea, o professor Massimo Benzi buscava por um meio fácil para os estudantes de design trabalhar com tecnologia. Na busca pela facilidade para trabalhar com tecnologia, Benzi discutiu seu problema com David Cuartielles, que era um pesquisador visitante da Universidade de Malmö, na Suécia, que procurava uma solução para um problema parecido (EVANS;NOBLE;HOCHENBAUM, 2013).

Benzi e Cuartielles decidiram criar um micro controlador que poderia ser utilizado pelos seus estudantes de artes e design nos projetos. Este micro controlador não poderia ser caro e que qualquer pessoa poderia utilizar. David Cuartielles desenhou uma placa, e um aluno de Massimo, David Mellis, programou um software para a placa. Gianluca Martino que trabalhou no *Design Institute*, engenheiro local, foi contratado pelo Massimo, assim concordou em produzir uma tiragem de duzentos placas. A placa criada foi nomeada *Arduino*, referência a um bar local frequentado por alunos e membros do corpo docente (EVANS;NOBLE;HOCHENBAUM, 2013).

2.2 NÚMEROS E ESTATÍSTICAS

Com a evolução da tecnologia, dispositivos foram ganhando melhorias, ganhando diversas novas utilidades, alguns com acesso à Internet, podendo gerar informações e enviando para um *Web Services.* Com esses dispositivos conectados à Internet define-se o termo *Internet of Things (IoT)*, segundo EVANS (2011).

Em 2015, havia cerca de 4,9 bilhões de dispositivos conectados, somente *smartphones* era 1,4 bilhões. Estima-se que em 2020 serão mais de 50 bilhões de dispositivos conectados, 250 milhões de carros sejam conectado com a Internet, gerando várias possibilidades diferentes de serviços (MARR, 2015).

A Figura 1 ilustra um breve infográfico a respeito das estatísticas de dispositivos conectados até 2020

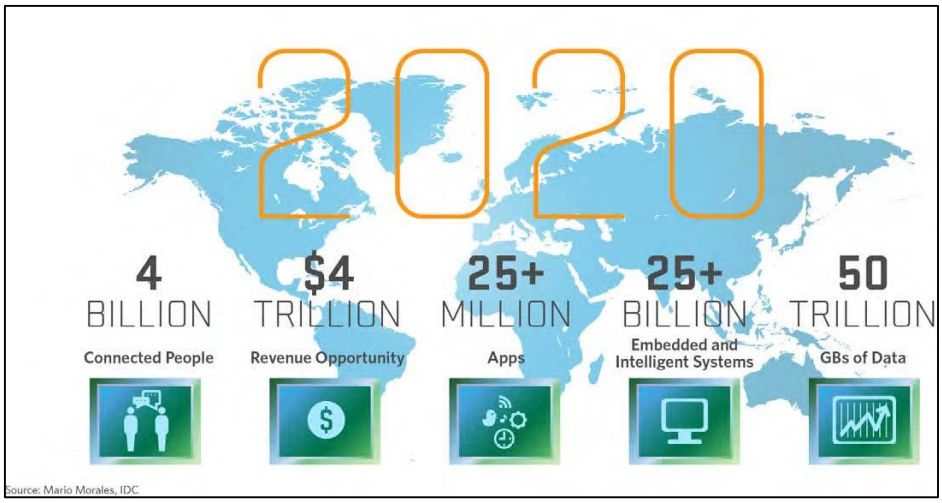

Figura 1: Dispositivos conectados até 2020.
Fonte: wordstream.com

A Google atualmente, contem carros conectados à Internet, dirigindo sozinhas pelo Estados Unidos, atualmente esses carros já dirigiram no total mais de 1,4 milhões de milhas (GOOGLE, 2016).

2.3 PLATAFORMAS COMPUTACIONAIS RELACIONADAS

2.3.1 Raspberry PI

O Raspberry Pi é um computador de baixo custo, onde apenas precisa de um sistema operacional, um monitor ou TV, e um kit de *mouse* e teclado. Por ser um computador, ele faz várias coisas, como navegar pela internet, assistir vídeos, edições de texto, jogar jogos, entre outros.

O Raspberry Pi pode interagir com outros componentes, igual ao Arduino, utilizando portas digitais, assim fazendo uma integração com componentes, como sensores.

Com a facilidade da utilização, ele é usado para a aprendizagem, tanto de programação, como da utilização de componentes assim podendo criar vários projetos, como robótica, sensores, casas inteligentes, entre outros (SJOGELID, 2013).

2.3.2 Intel Galileo/Edison

A plataforma Intel Galileo é uma placa desenvolvida utilizando um micro controladora com base Intel Quark SoC X1000, um sistema de 32 bits Intel Pentium. Essa plataforma tem uma facilidade de desenvolvimento, com suporte a sistemas operacionais como, Windows, Mac OS, Linux, utilizando o Arduino IDE para um ambiente de programação.

O Intel Galileo possui um slot mini-PCI Express, porta Ethernet para conexões de 100Mb, slot de cartão Micro-SD, entre outros recursos.

2.4 PRINCIPAIS COMPONENTES E SHIELDS

Os *Shields* placas auxiliares que fornecem novos sensores ao *Arduino* e expandem a capacidade de integrações com outros sistemas.

2.4.1 ARDUINO MOTOR SHIELD

Segundo Randy (2012), o *Arduino Motor Shield*, possibilita a integração de motores ao *Arduino*, facilitando a sua integração e manipulação, sem a necessidade de um conhecimento sobre vários componentes para realizar essa integração.

Na Figura 2 é ilustrado a placa *Motor Shield*.

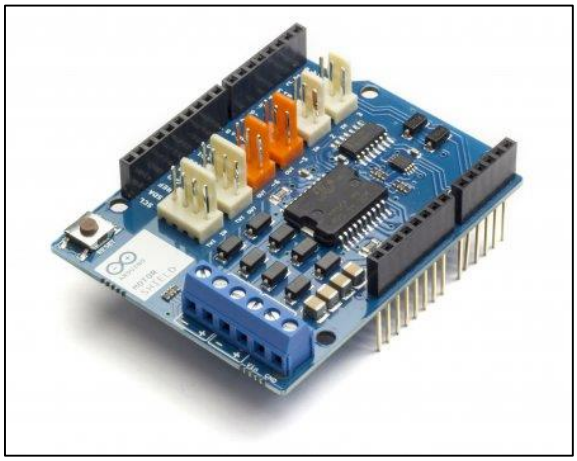

Figura 2: Arduino Motor Shield
Fonte: Arduino Store

2.4.2 STANDARD LCD 20x4

LCD 20x4 é um componente *LCD* para a exibição de informações do *Arduino*, necessitando de 5 volts de energia e 6 pinos de informação. Pequenas informações podem ser exibidas nele, assim abrindo diversas possibilidades para desenvolver projetos onde pouca informação deve ser entregue ao usuário, segundo INTOROBOTICS (2015).

Na Figura 3 é ilustrado o *Standard LCD 20x4*.

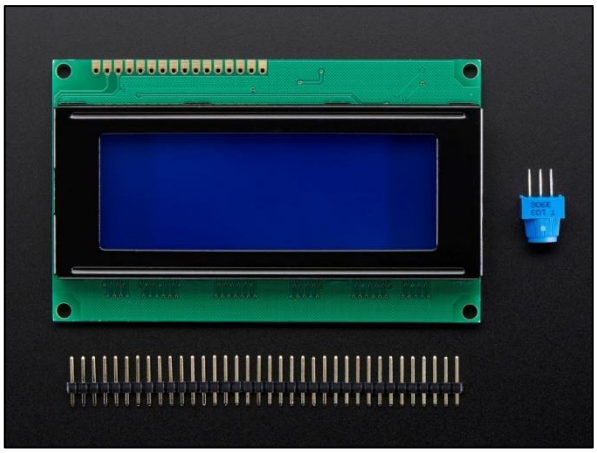

Figura 3: Standard LCD 20x4
Fonte: Adafruit

2.4.3 ETHERNET SHIELD

O *Ethernet Shield* auxilia a conectar o *Arduino* com a *Internet*, de modo simples, fácil e rápido. Encaixando ele no *Arduino*, e colocando o conector *RJ-45*, você já tem acesso à Internet, facilitando a comunicação para o usuário. Como vários *Shields* o hardware é *open-source*, liberando para outras pessoas podendo fazer o modulo, melhorando ou fazendo uma alternativa mais barata.

Em vários projetos, o *Ethernet Shield* é facilmente acoplado auxiliando no desenvolvimento do projeto, assim aumentando a capacidade do *Arduino* (GUTIÉRREZ, 2013).

Na Figura 4 é ilustrado o *Ethernet Shield*.

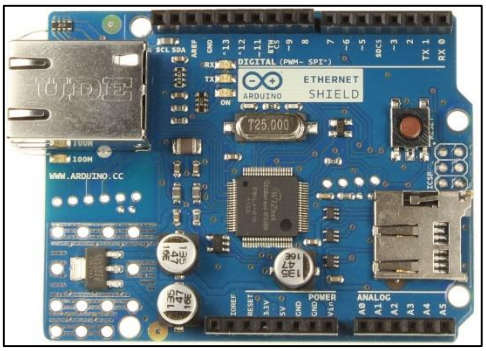

Figura 4: Ethernet Shield
Fonte: Arduino.cc

2.5 EXEMPLOS DE PROJETOS EM ARDUINO

Alguns projetos tem um grande destaque, auxiliando em vários setores, desde a agroindústria, construção civil, saúde, entre outros. Com a facilidade na programação e na aprendizagem sobre o *Arduino* e nos *Shields*, vários projetos cresceram de forma a alguns ganhar espaço.

O *Gas Leak Monitor*, é um projeto onde ele parte da ideia de um aplicativo que monitora a válvula de gás, assim protegendo construções para que não tenha um dano externo ou interno. A ideia começou quando foi verificado que de 2002 até 2012, foram registradas mais de 100 fatalidades com explosão de gás. Os canos com mais de 56 anos, custam entre 2 a 8 milhões de dólares por milha para realizar a manutenção (JASPER, 2016).

Outro projeto é o *IoT Home Security Model*, é um projeto para a segurança residencial, com sensores que verificam a residência, garantindo uma segurança extra, sendo controlada por dispositivos moveis e por um computador. As informações que são geradas pelos sensores, são constantemente armazenadas em um *cloud storage*, assim podendo ser manipuladas através desses dispositivos (KOW, 2016).

Na Figura 5 é ilustrado uma maquete utilizando o projeto *IoT Home Security Model*.

Figura 5: IoT Home Security Model
Fonte: www.hackster.io

2.6 ARDUINO IDE

O *Arduino IDE*, é um ambiente de desenvolvimento integrado que é utilizado para desenvolver os códigos que é utilizado pelo *Arduino*. O *Arduino IDE* utiliza a linguagem de programação *Java*, para a sua construção, onde o código fonte é disponível e encontrado para fazer *download*, para que outras pessoas possam contribuir.

A Figura 6 ilustra o *Arduino IDE*, exibindo a sua interface inicial.

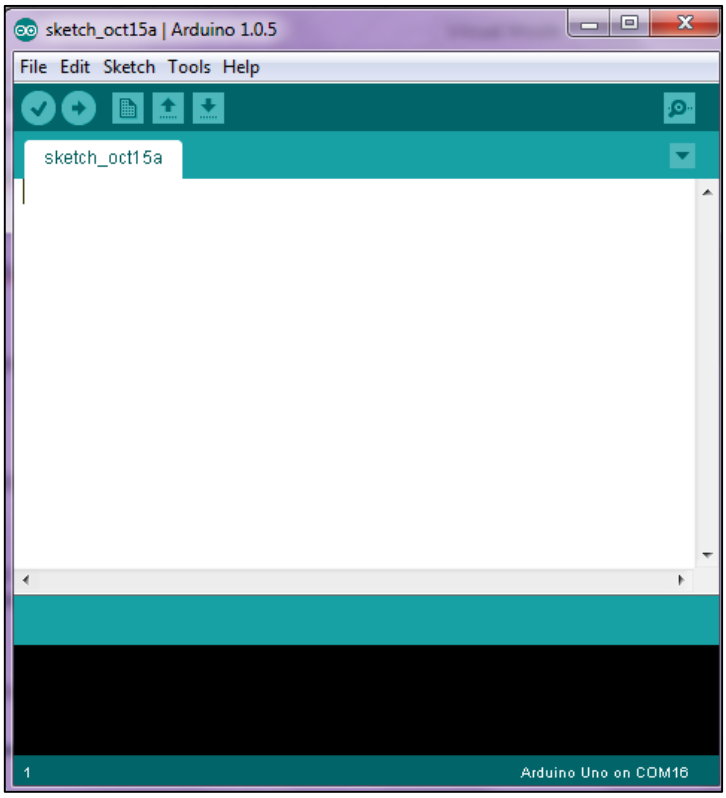

Figura 6: Arduino IDE
Fonte: LabDeGaragem

O *Arduino IDE* é um editor de código, que facilita para o desenvolvedor criar um código e enviar para o *Arduino*, podendo verificar se há erros no código e também ver e enviar dados. Os códigos escritos, é chamado de *sketches*, onde são salvos com a extensão ".ino" (SM, 2015).

2.7 PROJETO DE PISCAR LED

O projeto *blink*, é um projeto simples, que tem por objetivo ficar ligando e desligando um *LED*, que é utilizado no pino 13 do *Arduino*. Neste projeto, é utilizado alguns componentes: um *LED*, um resistor e uma placa *Arduino* (ARDUINO, 2015).

```
Int LED = 13;
void setup() {
  pinMode(LED, OUTPUT);
}

void loop() {
  digitalWrite(LED, HIGH);
  delay(1000);
  digitalWrite(LED, LOW);
  delay(1000);
}
```

Neste trecho de código é utilizado para o *blink*, onde a cada 1 segundo o *LED* irá acender e apagar. Os métodos *setup* e *loop* são obrigatórios para o *Arduino*.

O método *setup* é chamado apenas uma vez, quando o *Arduino* é energizado ou quando ele é reiniciado, onde ele utilizado para inicializar variáveis, modos dos pinos ou quando um trecho de código é utilizado uma única vez.

O método *loop* é chamado várias vezes enquanto o *Arduino* tiver energia, nele é encontrado os códigos que é utilizado sempre, como, por exemplo, ficar esperando uma mensagem ou ficar enviando uma mensagem.

Neste trecho é utilizado algumas funções que auxiliam a enviar informações para o *LED* para que ele possa acender e apagar, O método *pinModo* é utilizado dentro do método setup, onde é executado uma vez para definir se o pino vai ser de entrada de dados ou de saída, como no exemplo o pino do *LED* é utilizado o *OUTPUT* que é para saída de dados. O método *digitalWrite* que recebe dois parâmetros é utilizado para escrever um valor para algum pino, onde nesse exemplo é utilizado as constantes *HIGH* e *LOW*, que diz para o *LED* para acender ou apagar. Outro método utilizado é o *delay*, que tem a função de pausar o programa por um determinado tempo.

3 – CHROME APPS E CHROME EXTENSIONS

Neste capítulo serão exposto os conceitos necessários para o compreender o funcionamento do *Chrome Apps* e *Chrome Extensions*, bem como suas características e funcionalidades. Será feita uma introdução dos assuntos de forma a fornecer a base de conhecimento exigida pela área estuda.

As plataformas *Chrome Apps* e *Chrome Extensions* são um conjunto de funcionalidades e recursos utilizados na concepção e desenvolvimento de aplicações Web executados diretamente no navegador *Google Chrome*. As aplicações desenvolvidas com base no Chrome Apps e Chrome Extensions são capazes de realizar pequenas tarefas, como, por exemplo, uma calculadora, um visualizador dos *e-mails* não lidos, e entre outras atividades. Tais aplicações fornecem funcionalidades adicionais, bem como integração a outros sites ou serviços, melhorando a navegação pela Internet.

3.1 FUNCIONALIDADES E RECURSOS

Muitos aplicativos e extensões oferecem novas funcionalidades para o *Google Chrome*, desde maneiras rápidas de visualização de *e-mail* até como desbloqueios de *links*. Algumas dessas funcionalidades permitem que o usuário não perca tempo tendo que entrar em algum *site*, podendo utilizar aplicações para otimizar determinadas atividades, como já mencionado.

As aplicações e extensões contém recursos que auxiliam o *Google Chrome*, providenciando novas funções a ele, desde conectar um dispositivo por *bluetooth* e também na porta *serial*. Com o recurso *serial*, o *Google Chrome* consegue se conectar a dispositivos auxiliares, como o Arduino, possibilitando a leitura e manipulação de informações de sensores conectados ao mesmo, bem com o envio de dados da aplicação para o dispositivos embarcado. Dessa forma, a comunicação Serial possibilita a integração de aplicações desenvolvidas com Chrome Apps e Chrome Extensions a dispositivos embarcados.

3.2 MODELO E ELEMENTOS ARQUITETURAIS

Os aplicativos para o *Google Chrome* não são executados diretamente em uma aba do *browser*, mas sim executados em um *container* ou contexto separado da aplicação Chrome que, não necessariamente, exige uma conexão com a Internet. É importante ressaltar que há aplicações que obtém e enviam dados.

O *container* ou contexto é o ambiente de execução de uma aplicação Chrome. O aplicativo é semelhante a um conjunto de páginas da Internet contendo o documento *HTML*, *CSS* e *JavaScript*, porém ele não é executado no navegador, mas sim no *container*, não sendo executado diretamente no navegador.

A Figura 7 ilustra o modelo do *container* do aplicativo.

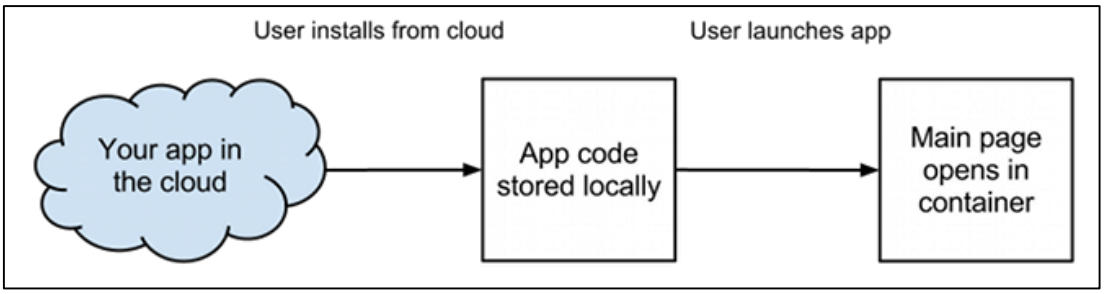

Figura 7: App Container Model
Fonte: developer.chrome.com

3.3 CICLO DE VIDA

O ciclo de vida das aplicações são estágios da execução, onde é gerenciada a maneira como as aplicações desenvolvidas para Chrome são iniciadas, executadas e finalizadas no *container* de execução.

O ciclo de vida de uma aplicação Chrome é gerenciado pelo *app runtime* e *event page*. O *app runtime* é responsável pela instalação do aplicativo, controlando o *event page* e podendo finalizar o aplicativo. Quando o *app runtime* é carregado, iniciando o *event page*, invocando o método *onLaunched()*, este método organiza o aplicativo com alguns parâmetros, como o de tamanho da tela.

Quando não há janelas abertas ou o aplicação é encerrado, o método *onSuspend()* é invocado, realizando pequenas tarefas de limpezas antes do container específico ser concluído e disponibilizado a outras aplicações.

Na Figura 8 ilustra como é o ciclo de vida de um *Chrome App*.

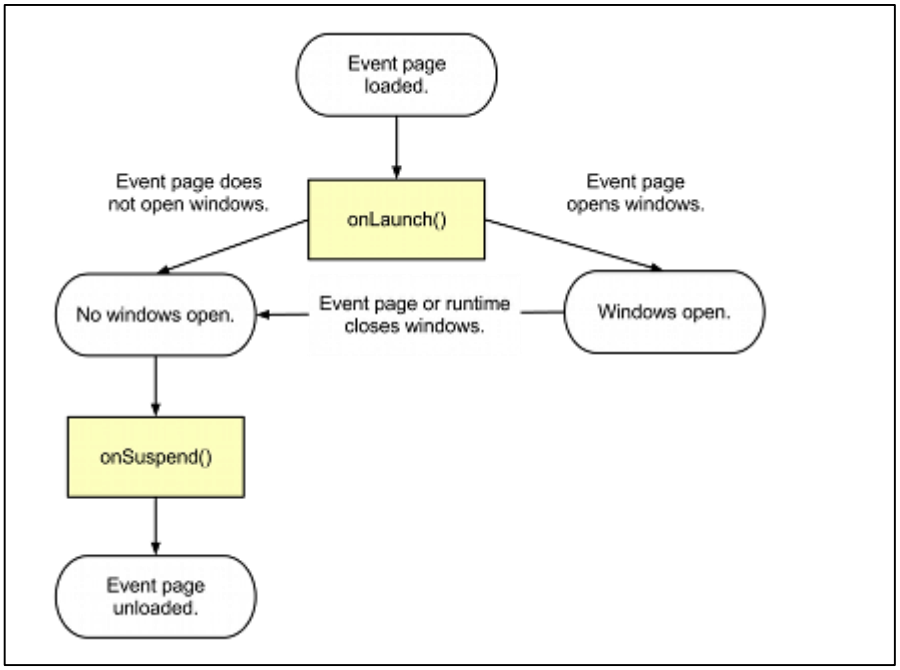

Figura 8: Chrome App Life Cycle
Fonte: developer.chrome.com

3.4 USO DE APPS E EXTENSIONS EM MOBILE

Segundo GRIVEVE (2014), o uso de *apps* e *extensions* no *Google Chrome*, trouxe bastante utilidade e velocidade a determinadas tarefas auxiliando em diversas áreas de aplicação. Pensando nisso, e no aumento da utilização de dispositivos *mobile*, com a programação baseada em *web*, os aplicativos que são desenvolvidos e utilizados em um *desktop*, podem ser utilizados também em dispositivos *mobiles*, assim o usuário não fica “preso” ao computador, podendo ter uma flexibilidade ao utilizar os *Apps* e *Extensions*.

Os aplicativos podem ser executados em dispositivos *mobiles*, nos sistemas operacionais *Android* e *iOS.* É feita a execução dos aplicativos em um conjunto de ferramentas baseadas no *Apache Cordova*, um framework de desenvolvimento *mobile* para construção de aplicativos com capacidade de utilização *HTML*, *CSS* e

JavaScript, ou seja, o *Apache Cordova* envolve o *app web* com um aplicativo, assim permitindo a distribuição no *Google Play*/*App Store* (GRIEVE, 2014).

Na Figura 9 é ilustrado um aplicativo na sua versão em *desktop* e na versão sendo executado em um *mobile*.

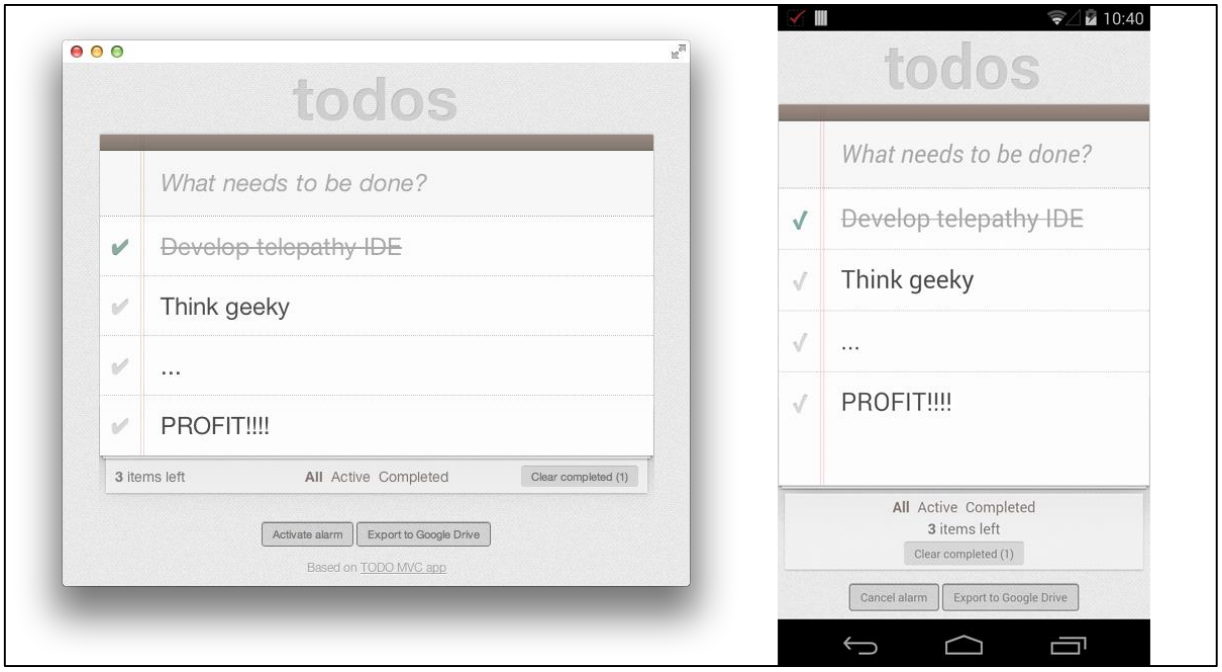

Figura 9: Aplicativo no *desktop* e *mobile*
Fonte: developer.chrome.com

O *Chrome Apps*/*Extensions* em um dispositivo *mobile* possui alguns recursos extras e também podendo ter acesso a *APIs* da plataforma *Cordova*, como *plugins* de acesso a câmera, liberando para tirar fotos ou escolher imagens da galeria.

4 – PROPOSTA DE TRABALHO

Neste capítulo apresenta-se a proposta de trabalho, onde o desenvolvimento de um aplicativo para a integração do Arduino com o *Google Chrome*, utilizando o *Chrome Apps* e *Chrome Extensions* para realizar essa integração.

O aplicativo desenvolvido foi implementado por meio de um editor de código que auxilia no uso do recurso do Chrome Apps, bem como do Chrome Extensions. Utilizando também o *Arduino IDE,* para desenvolver o código que enviará informações do *Arduino* para o *Google Chrome*.

O aplicativo apoia na integração do *Arduino* com o *Google Chrome*, facilitando a troca de informação e possibilitando que o usuário envie e receba dados extraídos de sensores conectados à plataforma Arduino.

4.1 OBJETIVOS E COMPONENTES DE INTEGRAÇÃO

Com diversos meios de comunicação entre o Arduino e a Internet, a aplicação Chrome proposta facilitará o envio e recebimento de dados para o usuário, assim não será preciso conhecer ou compreender códigos complexos, pois a aplicação fornecerá componentes que controlam as funcionalidades expostas.

Por meio da porta Serial, a plataforma *Arduino* envia informações para o computador, para que a aplicação possa tratar tais informações e executar as funcionalidades integradas ao projeto Arduino.

Para a integração desde projeto, foi necessário a utilização de um *Arduino* com alguns sensores para testes, como, alguns *LEDs* e um sensor de temperatura e umidade. Para a implementação, foi necessário a utilização da API *chrome.serial*, que irá receber os dados que o *Arduino* irá enviar, assim permitindo que o aplicativo use esses dados. Depois do projeto ser concluído, ele deve receber as informações dos sensores, exibindo-as, assim o usuário poderá analisar essas informações tomando as devidas providencias, enviando um comando, ou caso necessário, modificando fora do *Arduino*.

4.2 CENÁRIO PARA ADOÇÃO DA ABORDAGEM PROPOSTA

No desenvolvimento deste trabalho, implementa-se uma aplicação que interage e manipula o Arduino, liberando o acesso a dados dos sensores e atuadores para o usuário.

Para desenvolver este projeto, foi adotada a plataforma *Chrome Apps* e *Chrome Extensions*, utilizando a API *chrome.serial*, as informações geradas no Arduino são enviadas por meio da porta *serial*. Portanto, a API consumirá as informações e as disponibilizará para o usuário manipular.

Na Figura 10, ilustra um diagrama de como é feita a recepção dos dados, onde o Arduino conectado no computador, envia as informações pela porta *serial*, após a informação ser recebido pelo computador, a API *chrome.serial* captura a informação enviando para o aplicativo, assim o usuário juntamente com o aplicativo, manipula a informação, permitindo que seja feito uma ação ou apenas receber a informação.

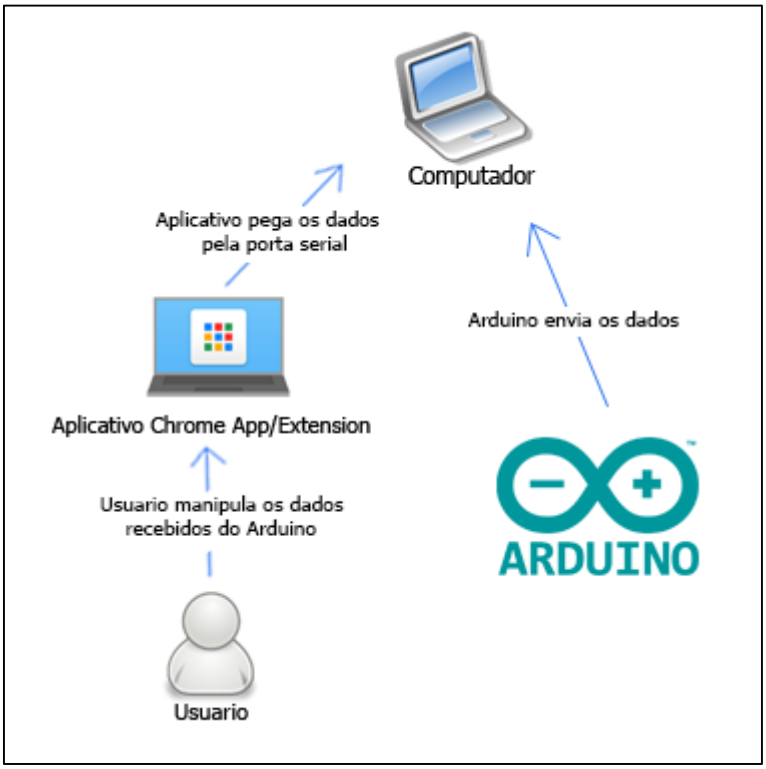

Figura 10: Diagrama Arquitetural

5 – DESENVOLVIMENTO DO TRABALHO

Neste capítulo, serão definidos a implementação. O aplicativo será desenvolvido para a plataforma *Chrome Apps* utilizando o *chrome.serial*, junto com o *Arduino* para o desenvolvimento de uma comunicação entre eles.

5.1 – IMPLEMENTAÇÃO DE PROJETO BASEADO EM CHROME APPS

A implementação da aplicação proposta adota os conceitos explorados nos capítulos anteriores sobre *Arduino* e a plataforma *Chrome Apps*.

As seções seguintes descrevem os componentes utilizados para o desenvolvimento deste projeto, bem trechos do código fonte resultantes da implementação.

5.1.1 – ARQUIVO *MANIFEST*

O arquivo *manifest.json*, é um arquivo que controla várias informações do aplicativo, desde o nome até a permissão que ele terá que ter para realizar certas operações.

Dentro do *manifest* alguns campos são obrigatório, como:

- *App,* nesse campo é onde coloca o *JavaScript* que irá abrir a tela principal
- *manifest_version,* neste campo é atribuído a versão do *manifest*, onde a última versão é a 2, que é utilizado para criar aplicativos baseado em Chrome Apps.
- *Name,* esse campo corresponde ao nome que o aplicativo terá.
- *Version*, version é a versão do aplicativo.

Demais campos podem ser colocados, sendo eles opcionais ou recomendados, como o *icons* um campo recomendado, pois caso seja publicado no *Chrome Web Store*, a aplicação deverá possuir um ícone oficial.

Nesta implementação foi utilizado a seguinte configuração:

```
{
  "name": "TCC",
  "version": "1",
  "manifest_version": 2,
  "permissions": ["serial"],
  "minimum_chrome_version": "36",
  "description": "App para comunicação com o Arduino",
  "icons": {
    "16": "assets/icon_16.png",
    "128": "assets/icon_128.png"
  },
  "app": {
    "background": {
      "scripts": ["js/background.js"],
      "transient": true
    }
  }
}
```

Neste arquivo, foram utilizados os campos obrigatórios e alguns opcionais e recomendados, como a *permissions*, que é responsável pelas permissões que a aplicação utiliza, neste caso foi utilizado o *serial*, que usará uma conexão com o *Arduino.*

5.1.2 – JAVASCRIPT UTILIZADO PELO CHROME APPS

A plataforma Chrome Apps utiliza um arquivo JavaScript que é utilizado para controlar a vida do aplicativo, onde nesse arquivo ele chama a página principal.

Cada aplicativo contém um arquivo JavaScript associado para que seja feita o controle de vida, onde dentro dele é informado algumas opções, como o tamanho do aplicativo, e o limite que ele pode ser alterado.

A implementação deste trabalho, foi utilizado esse arquivo apenas para controlar a altura inicial, o limite máximo e mínimo que ele pode ser alterado e também o arquivo *html* que será aberto inicialmente.

O trecho do arquivo JavaScript utilizado para implementar o trabalho:

```
chrome.app.runtime.onLaunched.addListener(function() {
  chrome.app.window.create('main.html', {
    id: "mainwin",
    bounds: {
      top: 0,
      left: 0,
      width: 500,
      height: 600
    },
          minWidth: 500,
    maxWidth: 500,
          minHeight: 600,
          maxHeight: 600
  });
})
```

Este trecho do arquivo inicia a execução do aplicativo, onde ele adiciona um *listener,* que é um ouvinte esperando a tela inicial ser criado, nele contém os parâmetros *width* que é a largura inicial, *height* que informa a altura inicial, também contém o *minWidth* e o *maxWidth* que informa a largura mínima e a máxima respectivamente, e também contém o *minHeight* e o *maxHeight* que informa qual a altura mínima e máxima que o aplicativo irá poder ser alterado.

5.1.3 – APLICAÇÃO PROPOSTA

A aplicação consiste em uma interface, onde o usuário pode se conectar ao *Arduino* para realizar algumas tarefas. Dentro dessas tarefas, contém o *monitor* que auxilia o usuário a visualizar as informações que chegam ao aplicativo do *Arduino*, podendo saber que tipo e como a informações é recebido pelo aplicativo. No *monitor*, consistem em uma caixa de texto, onde o usuário pode enviar determinada informação para o *Arduino*.

Na Figura 11 é ilustrada a interface da aplicação proposta:

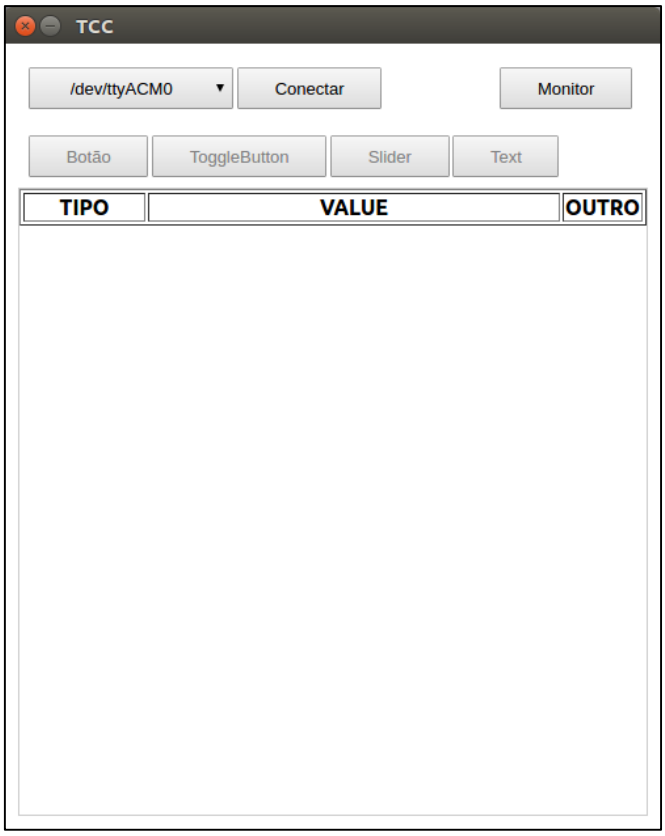

Figura 11: Tela Inicial.

Quando o botão *monitor* é pressionado, ele abre uma parte da tela para que seja exibido os componentes dele, assim o usuário possa ter controle sobre as informações que chegam do *Arduino*, e também tendo a possibilidade de enviar informações para testes rápidos.

Na Figura 12 é ilustrada a interface da aplicação proposta com o botão *Monitor* ativo:

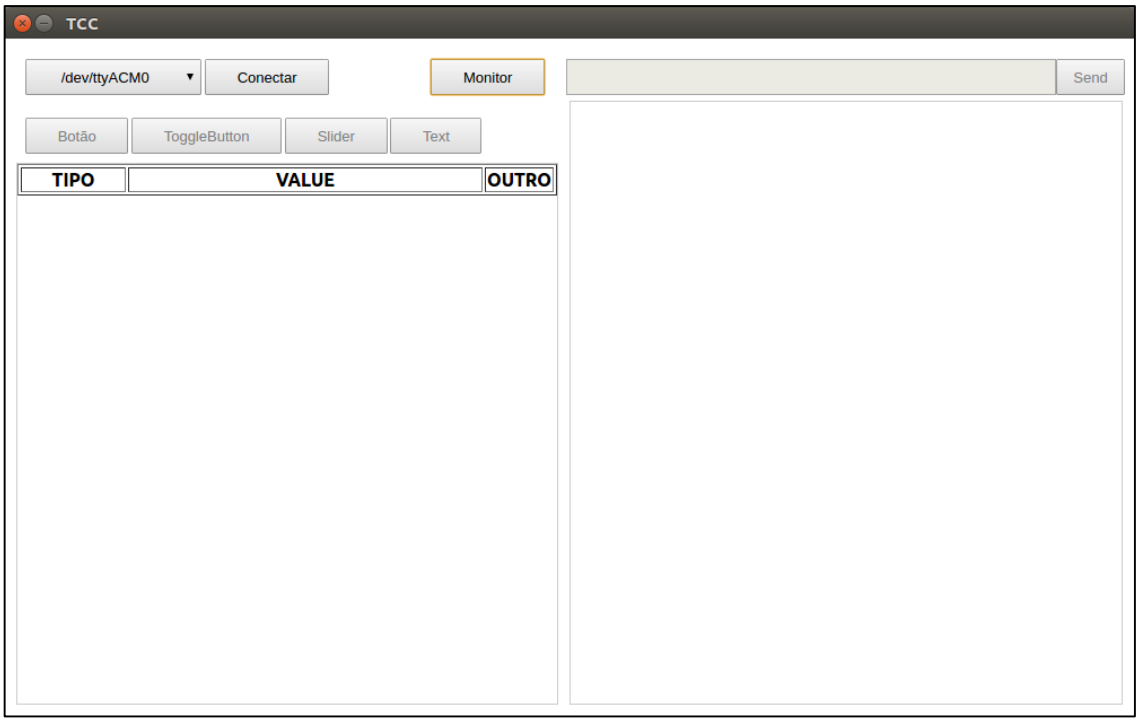

Figura 12: Tela Inicial com o monitor ativo.

O aplicativo contém quatro componentes: um *button*, um *toggle button*, um *slider*, e um *text*, onde cada um terá uma função:

- *Button*: Esse componente ele vai enviar uma determinada informação para o *Arduino.*
- *Toggle Button*: Este componente permite que o usuário envie 2 informação, uma para quando o componente estiver ligado e outra quando ele estiver desligado.
- *Slider*: Esse componente o usuário informa qual o valor mínimo, valor máximo e o pino que ele terá efeito, assim enviando para o *Arduino* uma informação conforme o usuário for alterando a sua base.
- *Text*: Este componente irá receber um determinado valor que o *Arduino* enviar, quando é utilizado ele, é informado uma *string* que será utilizado para determinar onde ele vai ser colocado na tela.

Na figura 13 é ilustrada a interface com os componentes adicionados e o *Arduino* já conectado para o envio e recebimento de dados pela porta *Serial*:

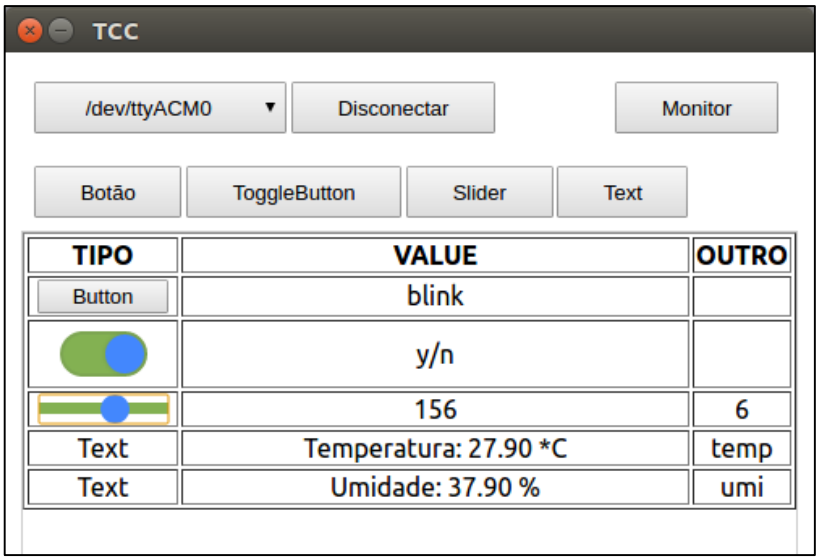

Figura 13: Tela inicial com componentes inseridos.

Na Figura 13 é ilustrada um diagrama onde o *Arduino* está conectado e recebendo e enviando dados para o aplicativo, onde o *button* enviará o texto *blink* quando ele for pressionado, já o componente *Toggle Button* irá enviar o texto *y/n*, caso ele estiver "verde" significa que está ativo assim enviando o texto de positivo, neste caso irá enviar o "y". O componente *Slider* enviar uma informação para o pino 6, com o valor dele. Os componentes *Text* serão utilizado para receber as informações de temperatura e umidade, onde eu coloco o texto que irá representar a informação desejada.

O trecho de código, mostra como foi feito para enviar as informações para o *Arduino* de um *button*:

```
$(document).on('click', '.button-send', function(e) {
        var buffer = str2ab($('#v' + e.target.id).html());
        chrome.serial.send(connectionId2, buffer, onSend);
});
```

Nesse trecho de código, mostra como os componentes *button, toggle button* e *slider* faz para enviar as informações para o *Arduino*, no caso do *button*, ele pega o *id* do botão que foi pressionado e pega o valor dele, enviando para o *Arduino* usando o *chrome.serial*.

No trecho a seguir, exibe como é enviado as informações usando um *toggle button*:

```
$(document).on('click', '.toggle-button', function(e){
	$('#' + e.target.id).toggleClass('toggle-button-selected');
	var values = $('#v' + e.target.id).html().split("/");
	var msg = (isSelected(e.target.id) ? values[0] : values[1]);
	var buffer = str2ab(msg);
	chrome.serial.send(connectionId2, buffer, onSend);
});
```

Este trecho de código, mostra como é enviado para o *Arduino* os valores do *toggle button*, ele pega o componente clicado e verifica o *id*, assim ele pega os valores e envia de acordo caso o *toggle button* estiver ligado ou desligado.

No trecho a seguir, exibe como é enviado as informações usando um *slider*:

```
$(document).on('change', '.position-input', function(e){
	$('#v' + e.target.id).html(this.value);
	var v = "sld:" + $('#vo' + e.target.id).html() + ":" + this.value;
	chrome.serial.send(connectionId2, str2ab(v), onSend);
});
```

Neste trecho de código, mostra como o *slider* envia as informações para o *Arduino*, quando o *slider* mudar de valor, ou seja, quando o usuário arrastar ele, este método irá pegar o valor e juntar com o pino em que ele está anexado, assim enviando para o *Arduino* usando o *chrome.serial*.

5.2 – AVALIAÇÃO EXPERIMENTAL DA ARQUITETURA PROPOSTA

Diante dos testes executados e dos resultados analisados, é possível analisar que a proposta deste trabalho foi realizada com todos os componentes funcionando, assim apresentando uma comunicação com o *Arduino*, recebendo e enviando informações.

Na figura 14 é ilustrada a interface da aplicação em funcionamento:

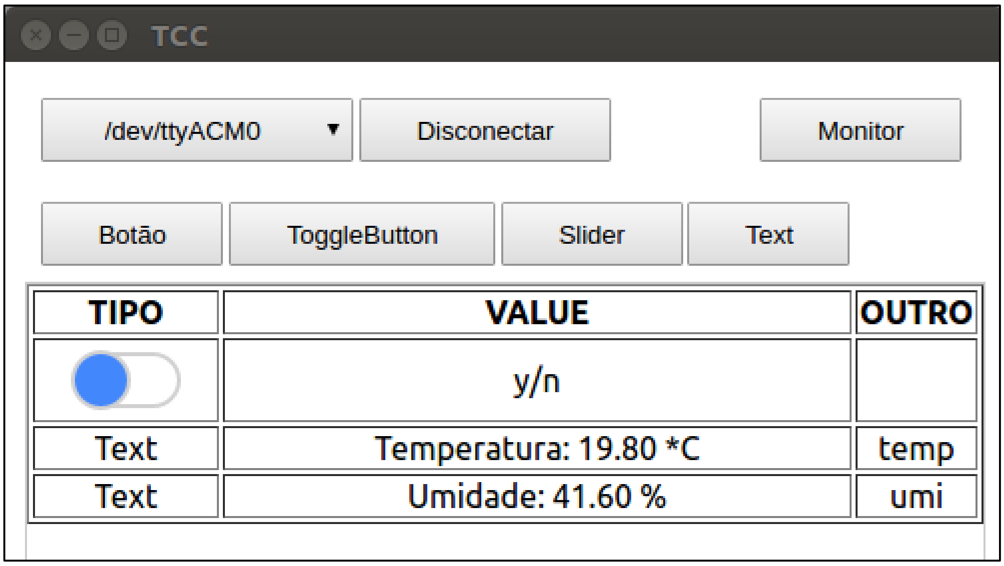

Figura 14: Tela do aplicativo

Na Figura 14, ilustra o aplicativo funcionando, onde exibe os campos *text* recebendo informações do *Arduino*, com as informações sobre "Temperatura" e "Umidade".

A Figura 15, ilustra para o usuário também o componente *monitor* que exibe todas as informações que estão, e também sendo possível enviar informações.

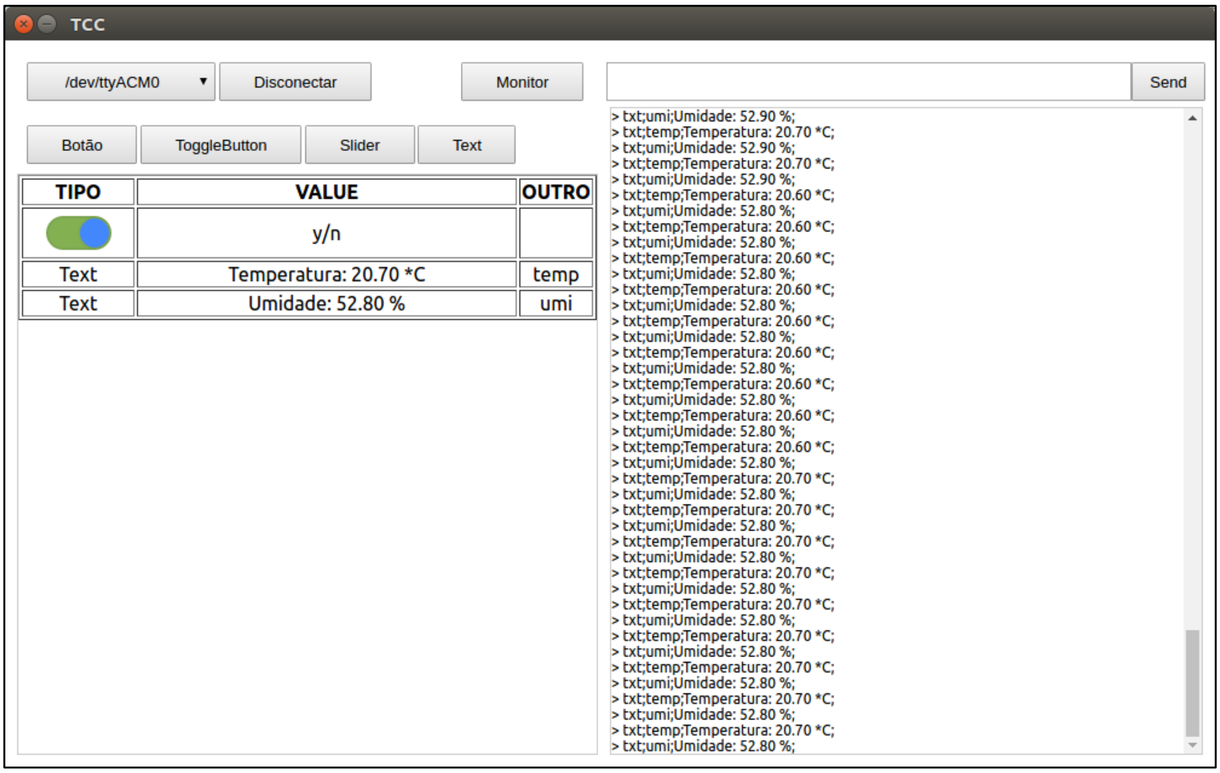

Figura 15: Tela inicial com o monitor aberto

6 – CONCLUSÃO

Neste capítulo serão descritos as considerações finais e as dificuldades do projeto, bem como trabalhos futuros são proposto tendo este como base, para continuação das pesquisas e implementações para um aplicativo que facilite a comunicação com o *Arduino*.

6.1 – CONSIDERAÇÕES FINAIS

O *Arduino,* pela facilidade e rapidez de utilização, torna-se um auxílio para o aprendizado e para uma prototipação simplificada. Utilizando o *Chrome Apps* como plataforma para o desenvolvimento da aplicação proposta para integração do Arduino na Web, pode-se abranger uma que facilite a comunicação entre o *Chrome Apps* e o *Arduino*.

Com a evolução das tecnologias, vários dispositivos móveis e embarcados têm sido desenvolvidos, fornecendo um auxílio para desenvolvedores e profissionais de tecnologia, desde de pequenas atividades, como acender uma luz, como grandes funções, como no auxílio da medicina. Muitos dispositivos utilizam recursos da *Internet* para enviar dados para outros dispositivos, assim apresentando uma comunicação entre dois ou mais dispositivos.

Com a ajuda das aplicações, as informações recebidas do *Arduino* acabam facilitando para que qualquer pessoa possa ler as informações e também enviar algumas configurações rápidas para o *Arduino*, assim controlando os sensores nele conectado.

Para desenvolver esse trabalho, algumas dificuldades foram encontradas, como a estrutura da interface. Assim podendo continuar o desenvolvimento do trabalho para chegar no resultado final.

6.2 – TRABALHOS FUTUROS

Com o conhecimento adquirido nesse projeto, muitas oportunidades de trabalho irão se abrir, desde conhecimento do *Arduino*, como eletrônica e programação para os componentes eletrônicos, e também conhecimento da plataforma *Chrome Apps*, levando o conhecimento do *HTML, CSS* e *JavaScript* usados pela plataforma *Chrome Apps* e *Chrome Extensions*.

Para um trabalho futuro, pode ser aprimorar este trabalho, acrescentando mais opções de componentes que podem ser utilizados para exibir e interagir com novos dados e sensores da plataforma Arduino. Também sugere-se agregar conceitos de *Web Services* para que as informações sejam enviadas para outras aplicações disponibilizadas na Web.

REFERÊNCIAS

ARDUINO. **Arduino Software (IDE)**. Disponível em < https://www.arduino.cc/en/Guide/Environment >. Acesso em: 29 jul. 2016.

ARDUINO. **Blink**. Disponível em <https://www.arduino.cc/en/Tutorial/Blink>. Acesso em: 29 jul. 2016.

BRAZILEIRO, Ricardo Borges. **tAMARINO: uma abordagem visual para prototipagem rápida em computação física**. 2013. 123p. Dissertação de Mestrado. Centro de Informática. Universidade Federal de Pernambuco, Pernambuco, Recife, 2013.

CHROME. **What Are Chrome Apps?**. Disponível em <https://developer.chrome.com/apps/>. Acesso em: 08 mar. 2016.

EVANS, Dave. **A Internet das Coisas: Como a próxima evolução da Internet está mudando tudo**. 1. Ed. Cisco Internet Business Solutions Group, 2011.

EVANS, Martin; Noble, Joshua; Hochenbaum, Jordan. **Arduino em Ação.** 1. Ed. Novatec Editora, 2013.

GEVA. **Paramecium**. Disponível em <http://making.do/paramecium/>. Acesso em: 18 out. 2015.

GOOGLE, **Google Self-Driving Car Project Monthly Report**. Disponível em <https://static.googleusercontent.com/media/www.google.com/pt-BR//selfdrivingcar/files/reports/report-0216.pdf>. Acesso em: 04 mar. 2016.

GRIEVE, Andrew. **Run Chrome Apps on mobile using Apache Cordova**. Disponível em <http://blog.chromium.org/2014/01/run-chrome-apps-on-mobile-using-apache.html>. Acesso em: 21 mar. 2016.

GUTIÉRREZ, José M. R., **Arduino + Ethernet Shield**. 2013. 42p.

INOVAÇÃO TECNOLÓGICA. **Luva-Sonar para cegos tem projeto aberto**. Disponível em <http://www.inovacaotecnologica.com.br/noticias/noticia.php?artigo=projeto-luva-sonar-cegos-arduino#.Vid6cfmrSM8/>. Acesso em: 21 out. 2015.

INTEL, **Board de Desenvolvimento Intel Galileo Gen 2**. Disponível em <http://www.intel.com.br/content/www/br/pt/embedded/products/galileo/galileo-overview.html>. Acesso em: 25 fev. 2016.

INTOROBOTICS, **The best Arduino LCD displays you can have right now**. Disponível em < https://www.intorobotics.com/the-best-arduino-lcd-displays-you-can-have-right-now/>. Acesso em: 29 jul. 2016.

JASPER, **Make a Gas Leak Monitor**. Disponível em <https://www.hackster.io/thejmeister/make-a-gas-leak-monitor-e2bc4f?ref=platform&ref_id=424_trending___&offset=15>. Acesso em: 02 mar. 2016.

KNOW, Aaron, **IoT Home Security Model**. Disponível em <https://www.hackster.io/aaronkow/iot-home-security-model-71e48e?ref=platform&ref_id=424_trending___&offset=102>. Acesso em: 04 mar. 2016.

LOMAS, Natasha. **BITalino**. Disponível em <http://techcrunch.com/2013/09/05/bitalino/>. Acesso em: 18 out. 2015.

MARR, Bernard. **17' Internet Of Things' Facts Everyone Should Read**. Disponível em <http://www.forbes.com/sites/bernardmarr/2015/10/27/17-mind-blowing-internet-of-things-facts-everyone-should-read/#74b832ab1a7a >. Acesso em: 04 mar. 2016.

MCROBERTS, Michael. **Arduino Básico**. 1. Ed. Tradução Rafael Zanolli. São Paulo: Novatec, 2011.

PIRES, Paulo F.; DELICATO, Flavia C.; BATISTA, Thais; BARROS, Thomaz; CAVALCANTE, Everton; PITANGA, Marcelo. **Plataformas para a Internet das Coisas**. 2015. 60 p.

PI, RASPBERRY. **Whats is Raspberry Pi**. Disponível em <https://www.raspberrypi.org/help/what-is-a-raspberry-pi/>. Acesso em: 25 fev. 2016.

RANDY. **Arduino Motor Shield Tutorial**. Disponível em <http://www.instructables.com/id/Arduino-Motor-Shield-Tutorial/>. Acesso em: 29 jul. 2016.

YEBAHI, Haissam. **InoSensor: Ferramenta para gerenciamento de sensors para Arduino**. 2013. 141p. Monografia de Pós-graduação. Universidade do Estado de Santa Catarina/UDESC: São Bento do Sul, 2013.