UNIVERSIDADE DO VALE DO SAPUCAÍ

GUILHERME SANCHES PEREIRA

JÉSSICA ADRIELE DO NASCIMENTO

GERENCIADOR DE CONSUMO DE ENERGIA ELÉTRICA RESIDENCIAL

POUSO ALEGRE, MG

2016

UNIVERSIDADE DO VALE DO SAPUCAÍ

GUILHERME SANCHES PEREIRA

JÉSSICA ADRIELE DO NASCIMENTO

GERENCIADOR DE CONSUMO DE ENERGIA ELÉTRICA RESIDENCIAL

Trabalho de Conclusão de Curso apresentado ao Curso de Sistemas de Informação da Universidade do Vale do Sapucaí como requisito parcial para obtenção do título de bacharel em Sistemas de Informação.

POUSO ALEGRE, MG

2016

LISTA DE FIGURAS

[Figura 1 - Modelo plataforma Arduino UNO REV 3. 11](#_Toc450073485)

[Figura 2 - *Raspberry* *Pi* 2 13](#_Toc450073486)

[Figura 3 - Modelo de Sensor de Corrente TC SCT 013-000 14](#_Toc450073487)

LISTA DE ABREVIATURAS E SIGLAS

AC Corrente Alternada

ANSI *American National Standards Institute*

API *Application Programming Interface*

CSS *Cascading Style Sheet*

HTML *HyperText Markup Language*

IDE *Integrated Development Environment*

IOT *Internet of Things*

LCD *Liquid Crystal Display*

REST *Representational State Transfer*

TC Transformador de Corrente

URI *Uniform Resource Identifier*

SUMÁRIO

1 QUADRO TEÓRICO 6

1.1 Internet das Coisas 6

1.2 Microcomputador, sensores e prototipação 7

1.2.1 Raspberry Pi 7

1.2.2 Arduino 8

1.2.3 Sensor de corrente TC 10

1.3 Linguagens de programação 11

1.3.1 NodeJS 11

1.3.2 JavaScript 11

1.3.3 Linguagem C 12

1.4 Frameworks e bibliotecas 12

1.4.1 AngularJS 13

1.4.2 Materialize 13

1.4.3 JQuery 13

1.4.4 Highcharts 14

1.4.5 Open Energy Monitor 14

1.5 Armazenamento e gerenciamento de dados 14

1.5.1 MySQL 14

1.5.2 MongoDB 15

1.6 Métodos de integração e suas tecnologias 15

1.6.1 RESTFUL 15

1.6.2 JavaScript Object Notation (JSON) 15

REFERÊNCIAS 17

# 1 QUADRO TEÓRICO

Para desenvolvimento do projeto, serão empregadas o uso de diversas tecnologias, ferramentas e conceitos que serão discutidos neste capítulo.

## 1.1 Internet das Coisas

O termo *Internet of Things* (Internet das Coisas) foi dito pela primeira vez, em 1999, por Kevin Ashton, cofundador do Auto-ID Center do *Massachusetts Institute of Technology* (MIT). Em artigo, Ashton (2009) alegou que a ideia original do termo *Internet* das Coisas previa a conexão entre todos os objetos físicos à internet, com capacidade de absorver informações por meio de identificação por radiofrequência e tecnologias de sensoriamento - as quais os permitiriam observar, identificar e compreender o mundo independentemente das pessoas e suas limitações de tempo, atenção e precisão.

A internet hoje, conta com quase três bilhões de usuários conectados segundo “*Global* *Inernet* *Report*” (KENDE, 2014). Segundo previsões de (STAMFORD, 2013), em 2020 o número de dispositivos conectados e interligados será de 26 bilhões. Algumas previsões mais otimistas, segundo a CISCO, prevê 50 bilhões de objetos conectados no mesmo período, gerando uma movimentação de mercado de US$ 14,4 trilhões até 2022 (EVANS, 2011).

Portanto segundo Lacerda e Lima-marques, (2015, p. 161):

As inovações que surgem no âmbito da Internet das Coisas ampliam o potencial humano em diversas áreas - tais como planejamento urbano (cidades, edifícios e transito inteligentes), meio ambiente (energia, água), indústria, comércio, turismo, educação, saúde, trabalho, segurança, programas sociais, governo - com soluções capazes de promover desenvolvimento econômico, sustentabilidade e qualidade de vida.

Tais fatos trazem para sociedade atual, uma demanda de pessoas, órgãos e empresas privadas, capazes de desenvolverem e programarem recursos de internet das coisas como forma de gerarem lucro financeiro e se inserirem em um mercado com ampla expansão. Este conceito de uma nova geração de conectividade, alinhado ao uso de novas tecnologias e ferramentas disponíveis no mercado, justificado pelo atual estado de crescimento tecnológico torna-se viável e eficaz o uso de recursos de *IoT* neste projeto.

## 1.2 Microcomputador, sensores e prototipação

O presente trabalho teve uma integração entre software e hardware resultando em uma solução composta de um microcomputador, uma plataforma de prototipação juntamente com sensores conectados a ela. Adiante está discriminado cada um desses componentes.

### 1.2.1 Raspberry Pi

“A *Raspberry Pi* é uma máquina completa, com considerável poder de processamento, em uma placa de circuito impresso menor do que um cartão de crédito” (UPTON e HALFACREE, 2013, p. 26).

Foi desenvolvida, em 2006, por Eben Upton, Rob Mullins, Jack Lang e Alan Mycroft, com base na Universidade de Laboratório de Informática de Cambridge e na empresa *Atmel ATmega*, que desenvolve placas com processadores ARM. Eben Upton e sua equipe, em 2009, oficialmente estabeleceram a *Raspberry Pi Foundation*, fundação de caridade educacional, com base no Reino Unido, que tem como meta, permitir que as pessoas de todas as idades possam aprender a programar e entender como funcionam os computadores. (RASPBERRY PI, 2014).

O coração do dispositivo *Raspberry Pi* é o processador multimídia *Broadcom* BCM2835, que tem a maioria dos componentes do sistema montado em um único e é baseado na arquitetura de conjunto de instruções ARM, desenvolvida pela *Acorn Computers*, no final dos anos 80, sendo ele, capaz de operar apenas com alimentação de energia de 1A e 5V, fornecida pela sua porta micro-USB. O baixo consumo do chip é traduzido, diretamente, em pouco desperdício de energia, mesmo durante tarefas complexas de processamento. Outra diferença importante entre o *Raspberry Pi* e seu desktop, sem falarmos no tamanho e no preço, é o sistema operacional – Linux, que possibilita fazer download de todo o código-fonte referente ao sistema operacional e realizar quaisquer alterações necessárias. (UPTON e HALFACREE, 2013).

Segundo Upton e Halfacree (2013), existem dois modelos da placa *Raspberry Pi*: Modelo A e Modelo B. A diferença entre eles é que o modelo B possui uma placa Ethernet e duas portas USBs, enquanto a modelo A, contém apenas uma porta USB e nenhuma Ethernet.

Nesse projeto foi utilizado o modelo *Raspberry Pi 2 model B+,* devido sua melhor configuração de *hardware* e aplicabilidade do projeto.

A Figura 2 apresenta o microprocessador *Raspberry Pi 2 Model B+*:

**Figura 2** **-** *Raspberry* *Pi* 2

******

**Fonte:** https://www.raspberrypi.org/wp-content/uploads/2015/01/Pi2ModB1GB\_-comp.jpeg

A *Raspberry Pi* recebe os dados coletados pela Arduino, apresentada na subseção 1.2.2 por meio dos sensores eletrônicos de corrente TC, que serão apresentados na subseção 1.2.3 e envia as informações para o banco de dados *online*.

### 1.2.2 Arduino

Arduino é uma plataforma de desenvolvimento e prototipagem *open-source* que, segundo Evans, Noble e Hochenbaum (2013, p.25), “a Arduino teve início no *Interaction* *Design* *Institute* de Ivrea, na Itália, em 2005”.

Esta plataforma teve sua criação destinada a ajudar estudantes de uma universidade italiana que tinha como premissa que: “[...] o preço almejado não poderia ser mais do que um estudante gastaria se saísse para comer uma pizza” (EVANS, et al., 2013, p.25).

Após sua criação, foram vendidas rapidamente as unidades fabricadas e a mesma começou a ser amplamente divulgada nas universidades da Itália e mais tarde pelo mundo todo.

Esta plataforma trabalha no conceito de entrada de dados, através de portas eletrônicas e/ou digitais, realiza-se um processamento dos dados e gera-se uma saída, sendo este procedimento executado em *loops* predefinidos pela linguagem de programação.

A linguagem utilizada nessa plataforma é a linguagem C, que tornou-se uma linguagem adaptável a qualquer dispositivo que siga o padrão ANSI definido para a linguagem.

Segundo o site oficial da plataforma, no que diz respeito a evolução de sua criação:

Ao longo dos anos, Arduino tem sido o cérebro de milhares de projetos, desde objetos do cotidiano para instrumentos científicos complexos. A comunidade mundial de tomadores - estudantes, amadores, artistas, programadores e profissionais - reuniu em torno desta plataforma open-source, suas contribuições acrescentaram-se a uma quantidade incrível de conhecimento acessível que pode ser de grande ajuda para novatos e especialistas similares (ARDUINO, 2016, p.1).

O *hardware* que compõe a plataforma Arduino é variado conforme os diversos modelos que foram lançados desde seu lançamento em 2005. Para este projeto iremos analisar o modelo UNO REV 3:

Figura - Modelo plataforma Arduino UNO REV 3.



**Fonte**: http://www.embarcados.com.br/wp-content/uploads/2013/11/imagem\_01.png.

Como visto na Figura 1, a Arduino é composta por:

* 14 pinos de entrada e saída digital (pinos 0-13):

Esses pinos podem ser utilizados como entradas ou saídas digitais de acordo com a necessidade do projeto.

* 6 pinos de entradas analógicas (pinos A0 - A5):

Esses pinos são dedicados a receber valores analógicos, por exemplo, a tensão de um sensor. O valor a ser lido deve estar na faixa de 0 a 5 V que serão convertidos para valores entre 0 e 1023.

* 6 pinos de saídas analógicas (pinos 3, 5, 6, 9, 10 e 11):

São pinos digitais que podem ser programados para ser utilizados como saídas analógicas.

A alimentação da placa pode ser feita a partir da porta USB do computador ou através de um adaptador AC. Para o adaptador AC recomenda-se uma tensão de 9 a 12 volts. Juntamente na placa existe um processador da família Atmel que muda em cada modelo de Arduino.

Para se desenvolver nesta plataforma, é necessário o uso de um *software* que faça *upload* do código programado em linguagem C para dentro do micro controlador embutido na placa e que o mesmo seja executado quando a Arduino for conectada à rede elétrica. Segundo o site da equipe Embarcados (2016) para se programar na Arduino é necessário a utilização de uma IDE que permite a criação de um código para a placa e após a escrita do código deve-se clicar sobre o botão de *upload* da IDE que então irá traduzir este código feito em linguagem C para uma outra linguagem que possa ser compreendida pelo micro controlador.

Neste projeto a Arduino é responsável pela conectividade com os sensores que realizam a leitura da energia elétrica e disponibilizam estes dados para a central *Raspberry* conectada à nuvem. Sua facilidade de aprendizado e compra por um baixo custo foram fatores relevantes na escolha dessa plataforma.

### 1.2.3 Sensor de corrente TC

Os sensores de correntes TC são dispositivos eletrônicos desenvolvidos para serem aplicados em diversos circuitos elétricos, através de variadas plataformas existentes de prototipagem (Arduino PIC, *Raspberry*) para mensurarem a corrente elétrica de algum dispositivo. Ele é conectado diretamente à Arduino e faz a mensuração dos valores de corrente de cada cômodo/disjuntor enviando os dados para a plataforma de desenvolvimento.

Existem diversos modelos no mercado com diversas potências máximas de mensuração. Para o presente projeto está sendo utilizado modelos de 100 amperes.

**Figura 3** - Modelo de Sensor de Corrente TC SCT 013-000



**Fonte:** http://s3.amazonaws.com/img.iluria.com/product/18B9DA/3A3A8A/450xN.jpg

Os sensores da família TC SCT são denominados não-invasivos, ou seja, como pode ser visto na Figura 3, ele detém de um encaixe lateral que se abre e fecha para a colocação do fio que precisa ser medido a corrente, este fato faz com que não se necessite qualquer manuseio da fiação oferecendo maior segurança na sua instalação.

## 1.3 Linguagens de programação

O desenvolvimento de um software é baseado em uma escrita de códigos em uma linguagem de programação específica. Dentre as mais variadas linguagens presentes e disponíveis na comunidade tecnológica foi utilizada as linguagens descritas abaixo.

### 1.3.1 NodeJS

Segundo Pereira (2015), NodeJS foi lançado em 2009 por Ryan Dahl com mais 14 colaboradores para solucionar um problema de bloqueio de execução em sistemas com muitas requisições simultâneas, isso fazia com que um processo em execução travava os outros até que se terminasse a execução do mesmo e assim os demais viessem a ser executados em fila.

“Esta tecnologia possui um modelo inovador, sua arquitetura é totalmente *non*-*blocking* *thread* (não-bloqueante), apresentando uma boa *performance* com consumo de memória e utilizando ao máximo e de forma eficiente o poder de processamento dos servidores[...]” (PEREIRA, 2015, p.2).

Esta tecnologia foi utilizada no desenvolvimento do serviço *RestFul,* sendo implementada no *back-end* do projeto.

### 1.3.2 JavaScript

*JavaScript* é uma linguagem de programação criada pela Netscape em parceria com a Sun Microsystems. Sua primeira versão, definida como *JavaScript* 1.0, foi lançada em 1995 e implementada em março de 1996 no navegador Netscape Navigator 2.0 (SILVA, 2010).

Conforme Mozilla Developer Network (2016), *JavaScript* é baseado na linguagem de programação ECMAScript, o qual é padronizado pela Ecma International na especificação ECMA-262 e ECMA-402.

*JavaScript* está sendo utilizado no *core* aplicação *web* que contém a interface que o cliente/usuário acessa para verificação de todos seus dados lidos pelos sensores. A utilização de um *framework* *JavaScript* também foi utilizado e detalhado logo abaixo, nas próximas seções.

### 1.3.3 Linguagem C

A linguagem C foi criada em 1972 em um laboratório chamado *Bell Telephone Laboratories*, por Dennis Ritchie, para que fosse permitido a escrita de um sistema operacional, o UNIX, que utilizasse os benefícios de uma linguagem de alto nível, deixando de lado o uso de linguagens de baixo nível que detinham de uma maior complexidade para seu uso (DAMAS, 2007).

Inicialmente, a linguagem servia para desenvolvimento de programas de sistemas que são, segundo Schildt (1986, p.2), “[...] uma classe de programa que, ou são parte, ou operam em conjunto com o sistema operacional do computador [...]”.

Atualmente, após mais de quatro décadas de sua criação, a linguagem se apresenta ao mercado também em dispositivos eletrônicos de pequeno porte como micro controladores pelo fato de ser uma linguagem de fácil portabilidade. Tal expansão da linguagem pode ser observada em Klotz (2010, p.3) que diz que “uma das melhores características da linguagem C é que não está ligada a qualquer equipamento ou sistema em particular. Isto torna mais fácil para um usuário para escrever programas que serão executados sem quaisquer alterações em praticamente todas as máquinas”[[1]](#footnote-1).

Devido a todos estes conceitos teóricos e históricos no que tange a abrangência da linguagem C em dispositivos embarcados, torna-se viável o uso da mesma no projeto, atuando na plataforma de desenvolvimento embarcado, o Arduíno, fazendo toda a leitura da corrente elétrica e enviando os dados para uma central receptora, que é outra plataforma para desenvolvimento, a *Raspberry.*

## 1.4 Frameworks e bibliotecas

Para obter o resultado esperado com maior rapidez e qualidade, optou-se por utilizar algumas bibliotecas e frameworks. As bibliotecas são empregas para, na maioria nas ocasiões, prover uma simplificação de criação de código nas etapas do desenvolvimento, já que as mesmas proveem uma gama de funções e métodos pré-definidos, poupando assim, o esforço do desenvolvedor em criar todo código a partir do zero. Diferentemente das bibliotecas, em que adequamos seu uso dentro do código, em frameworks temos a adequação do código dentro do framework, e nesse contexto temos então diversos recursos disponíveis nos mais variados frameworks existentes na comunidade. Confira adiante as especificações das bibliotecas e frameworks utilizados no trabalho.

### 1.4.1 AngularJS

*AngularJS* é um framework *open*-*source* mantido pela Google. Seu objetivo é estruturar o desenvolvimento *front*-*end* utilizando o modelo de arquitetura *model*-*view*-*controller* (MVC). O *framework* associa os elementos do documento HTML com objetos JavaScript, facilitando a manipulação dos mesmos.

Atualmente, existem duas versões do *framework,* a primeira e mais amplamente conhecida é a que foi utilizada neste projeto, enquanto a outra lançada no ano de 2016 contempla a utilização de outra forma de desenvolvimento que utiliza o *TypeScript*, que nada mais é que uma abstração da linguagem *JavaScript* para facilitar o desenvolvimento e se assemelhar à outras linguagens de programação como Java por exemplo, através da criação de componentes, existência de herança e variáveis com tipos definidos.

### 1.4.2 Materialize

Um *framework* *front-end* com *layout* moderno e responsivo que segue os padrões de desenvolvimento baseado em *Material* *Designer* da Google (MATERIALIZE, 2016).

Criado e projetado pela Google, o *Material* *Designer* é a combinação de princípios clássicos de projetos bem-sucedidos que contempla inovação e tecnologia, seguindo o objetivo de unificar a experiência do usuário em todos os seus produtos independente de qual plataforma está sendo utilizada (MATERIALIZE, 2016).

Neste projeto foi implementado em toda aplicação que possui acesso direto dos usuários. Seu uso favorece a usabilidade do cliente que contará com uma interface totalmente responsiva e inovadora.

### 1.4.3 JQuery

*JQuery* é uma biblioteca baseada em *JavaScript* de *software* livre e aberto criado por John Resig e seu uso está sob as licenças do MIT e GPL, ou seja, é permitido que qualquer pessoa ou empresa use de seus recursos para fins pessoais como também profissionais (SILVA, 2014).

Em 2005 o autor começou a elaborar suas primeiras intenções sobre como resolver problemas de robustez de código para solução de simples problemas (SILVA, 2014). Foi então que em 2006 foi lançado o primeiro *plugin* para a biblioteca e disponibilizada de forma pública.

Ainda segundo Silva (2014), a função dessa biblioteca é adicionar interatividade e dinamismo às páginas *web* que consequentemente irão proporcionar ao desenvolvedor, mecanismos necessários à criação de *scripts* fácil e usável, sem obstrução e causando no usuário final uma melhor experiência de usabilidade na web.

Essa biblioteca segue os padrões de desenvolvimento *web* e é totalmente compatível com os novos recursos do atual CSS3. Porém, deve-se atentar em seu uso para que esteja nas conformidades das normas do W3C.

Seu uso no projeto tem como premissa a disponibilização de maior rapidez e usabilidade no desenvolvimento do código *front-end* que é destinado à aplicação para uso do cliente final da aplicação *web* proposta.

### 1.4.4 Highcharts

### 1.4.5 Open Energy Monitor

## 1.5 Armazenamento e gerenciamento de dados

Para se obter a persistência de dados dentro de um software é necessário armazená-los dentro de algum banco de dados, seja ele relacional, orientado a documentos, orientado a grafos ou outros. Este armazenamento é algo fundamental para qualquer solução web e neste trabalho foi utilizado dois tipos de banco de dados, cada um para armazenar dados específicos ao qual a informação é necessária ser persistida. Adiante temos suas especificações e referencial teórico.

### 1.5.1 MySQL

MySQL é o banco de dados de código aberto mais popular do mundo e possibilita a entrega econômica de aplicativos de banco de dados confiáveis, de alto desempenho e redimensionáveis, com base na *web* e incorporados (ORACLE, 2016).

Além da facilidade de uso, do alto desempenho e da confiabilidade do MySQL, pode-se beneficiar dos recursos avançados, das ferramentas de gerenciamento e do suporte técnico para desenvolver, implementar e gerenciar seus aplicativos.

É amplamente utilizado no projeto para armazenar os dados lidos e todas as demais configurações e informações que necessitam de serem persistidas de forma *online,* como contas de usuário, históricos de acessos e outros.

### 1.5.2 MongoDB

## 1.6 Métodos de integração e suas tecnologias

Para atender aos requisitos do projeto necessitou-se o uso de uma forma de desacoplar as camadas envolvidas na solução, afim de facilitar futuras alterações e manutenção do software. Estas tecnologias e conceitos são descritas abaixo.

### 1.6.1 RestFul

Em 2000, Roy Fielding, em sua tese de doutorado apresentou uma nova forma de integrar sistemas hipermídias distribuído chamado REST (*Representational State Transfer*) (SANTOS, 2009).

Ainda de acordo com Santos (2009), REST é um estilo de arquitetura de software para sistemas hipermídia distribuídos, no qual utilizamos um navegador web para acessar recursos, mediante a digitação de uma URL.

Esta arquitetura e conceito foi aplicada no core da API[[2]](#footnote-2) para padronizar e desacoplar nossas camadas de desenvolvimento que são: hardware, serviço e aplicação web. Esta arquitetura nos traz manutenibilidade do sistema e benefícios para futuras mudanças na arquitetura de hardware.

### 1.6.2 JavaScript Object Notation (JSON)

JSON é um formato de dados que pode ser trocado, permutado e que se assemelha nos padrões de *scripts,* porém, não é considerado um subtipo da sintaxe do *JavaScript* (MOZILLA DEVELOPER NETWORK, 2014).

“ JSON é capaz de representar números, booleanos, textos, listas (sequência ordenada de valores) e objetos (mapeamento de valores de texto) composto por estes valores (ou por outras listas e objetos) ” (MOZILLA DEVELOPER NETWORK, 2014, p. 1).

Um JSON está constituído em duas estruturas:

* Uma coleção de pares nome/valor.
* Uma lista ordenada de valores.

A aplicação deste formato na solução é realizada para desacoplarmos nossas entidades de *hardware*, serviço e aplicação *web,* pois este formato é entendido facilmente por seres humanos e traduzidos com a mesma facilidade nos ambientes de máquinas, ou seja, podemos trocar totalmente qualquer entidade citada anteriormente por outra, até mesmo com outros equipamentos ou outras linguagens, bastando mantermos o formato JSON de entrada e saída de dados.

# REFERÊNCIAS

ARDUINO Introduction. **Arduino**, 2016. Disponivel em: <https://www.arduino.cc/en/Guide/Introduction>. Acesso em: 10 Março 2016.

ASHTON, K. That "Internet of Things" Thing. **RFID Journal**, 2009. Disponivel em: <http://www.rfidjournal.com/articles/view?4986>. Acesso em: 20 Março 2016.

DAMAS, L. **Linguagem C**. 10. ed. Rio de Janeiro: LTC, 2007.

EVANS, D. **The internet of things:** how the next evolution of the internet is changing everything. [S.l.]: CISCO white paper, 2011. 11 p.

EVANS, M.; NOBLE, J.; HOCHENBAUM, J. **Arduino em ação**. São Paulo: Novatec Editora, 2013.

GIL, A. C. **Como elaborar projetos de pesquisa**. 4. ed. São Paulo: Atlas, 2007.

INTRODUÇÃO ao Angular JS. **DEVMEDIA**, 2016. Disponivel em: <http://www.devmedia.com.br/introducao-ao-angularjs-curso-completo-de-angularjs-aula-1/32148>. Acesso em: 10 Março 2016.

KENDE, M. Global Internet report. **Geneva:** Internet Society, 2014. Disponivel em: <http://www.internetsociety.org/doc/global-internet-report>. Acesso em: 21 Março 2016.

KIOSKEA. Condução de reunião. **CCM**, 2014. Disponivel em: <http://br.ccm.net/contents/gestao-de-projeto-1343593641#579>. Acesso em: 02 Maio 2016.

KLOTZ, D. C for Embedded Systems Programming. **NXP**, 11 Novembro 2010. Disponivel em: <http://www.nxp.com/files/training/doc/dwf/AMF\_ENT\_T0001.pdf>. Acesso em: 10 Março 2016.

LACERDA, F.; LIMA-MARQUES, M. Da necessidade de princípios de Arquitetura da Informação para a Internet das Coisas. **Perspectivas em Ciência da Informação**, Belo Horizonte, 20, Junho 2015. 158-171.

MATERIALIZE. Documentação - Materialize. **Materialcss**, 2016. Disponivel em: <http://materializecss.com/>. Acesso em: 21 Março 2016.

MOZILLA DEVELOPER NETWORK. **JSON**, 2014. Disponivel em: <https://developer.mozilla.org/pt-BR/docs/JSON>. Acesso em: 20 Março 2016.

MOZILLA DEVELOPER NETWORK. **JavaScript**, 2016. Disponivel em: <https://developer.mozilla.org/pt-BR/docs/Web/JavaScript>. Acesso em: 22 Abril 2016.

ORACLE. MySQL | O Banco de Dados de Código Aberto Mais Popular. **ORACLE**, 2016. Disponivel em: <http://www.oracle.com/br/products/mysql/overview/index.html>. Acesso em: 20 Fevereiro 2016.

PEREIRA, C. R. **Node.js:** Aplicações web real-time com Node.js. São Paulo: Casa do Código, 2015.

RASPBERRY PI, 2014. Disponivel em: <https://www.raspberrypi.org/help/what-is-a-raspberry-pi/>. Acesso em: 20 Março 2016.

SANTOS, W. R. D. **Univale**, 2009. Disponivel em: <http://www.univale.com.br/unisite/mundo-j/artigos/35RESTful.pdf>. Acesso em: 23 Fevereiro 2016.

SCHILDT, H. **Linguagem C:** Guia do Usuário. São Paulo: MC Graw Hill, 1986.

SILVA, M. S. **JavaScript:** Guia do Programador. São Paulo: Novatec Editora, 2010.

SILVA, M. S. **HTML5:** A linguagem de marcação que revolucionou a web. São Paulo: Novatec Editora, 2011.

SILVA, M. S. **CSS3:** Desenvolva aplicações web profissionais com uso dos poderosos recursos de estilização das CSS3. São Paulo: Novatec Editora, 2012.

SILVA, M. S. **JQuery:** A Biblioteca do Programador JavaScript. 3. ed. São Paulo: Novatec Editora, 2014.

SOUZA, F. Primeiros Passos. **Arduino**, 2013. Disponivel em: <http://www.embarcados.com.br/arduino-primeiros-passos/>. Acesso em: 22 Fevereiro 2016.

SOUZA, F. Arduino UNO - Conheça os detalhes de seu hardware. **Embarcados**, 2016. Disponivel em: <http://www.embarcados.com.br/arduino-uno/>. Acesso em: 20 Fevereiro 2016.

STAMFORD, C. Gartner Says the Internet of Things installed base will grow to 26 billion units by 2020. **Gartner**, 2013. Disponivel em: <http://www.gartner.com/newsroom/id/2636073>. Acesso em: 21 Março 2016.

TARTUCE, T. J. A. **Métodos de pesquisa**. Fortaleza: UNICE -Ensino Superior, 2006.

TAURION, C. **Software embarcado:** oportunidades e potencial de mercado. Rio de Janeiro: Brasport, 2005.

TRAVASSOS, G. Internet das Coisas- Expectativas e desafios em 2015. **STARTUPI**, 2015. Disponivel em: <http://startupi.com.br/2015/01/internet-das-coisas-expectativas-e-desafios-em-2015/>. Acesso em: 30 Março 2016.

UPTON, E.; , G. H. **Raspberry pi:** Manual do usuário. Tradução de Celso Roberto Paschoa. São Paulo: Novatec Editora, 2013.

1. Original “One of the best features of C is that it is not tied to any particular hardware or system. This makes it easy for a user to write programs that will run without any changes on practically all machines”. Traduzido pelos autores. [↑](#footnote-ref-1)
2. API - *Application Programming Interface* (Interface de Programação de Aplicativos). É a interface, normalmente documentada que uma biblioteca ou framework disponibiliza para o programador possa utilizá-la. [↑](#footnote-ref-2)