UNIVERSIDADE FEDERAL DO CEARÁ

CAMPUS QUIXADÁ

BACHARELADO EM SISTEMAS DE INFORMAÇÃO

EMMANUEL CAINÃ ALVES DE MELO

SALAS DE AULA INTELIGENTES: UTILIZAÇÃO DE ARDUINO E BLUETOOTH LOW ENERGY COMO BEACONS PARA O MAPEAMENTO DE SALAS DE AULA

QUIXADÁ – CEARÁ

2016

EMMANUEL CAINÃ ALVES DE MELO

SALAS DE AULA INTELIGENTES: UTILIZAÇÃO DE ARDUINO E BLUETOOTH LOW ENERGY COMO BEACONS PARA O MAPEAMENTO DE SALAS DE AULA

Monografia apresentada no curso de Sistemas de Informação da Universidade Federal do Ceará, como requisito parcial à obtenção do título de bacharel em Sistemas de Informação. Área de concentração: Computação.

Orientador: Prof. Dr. Marcio Espíndola Freire Maia

QUIXADÁ – CEARÁ

2016

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Biblioteca Universitária
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

M485s Melo, Emmanuel Cainã Alves de.
Salas de aula inteligentes : utilização de arduino e bluetooth low energy como beacons para o mapeamento de salas de aula / Emmanuel Cainã Alves de Melo. – 2016.
55 f. : il. color.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Campus de Quixadá, Curso de Sistemas de Informação, Quixadá, 2016.
Orientação: Prof. Dr. Marcio Espíndola Freire Maia.

1. Internet das coisas. 2. Bluetooth. 3. Computação móvel. I. Título.

CDD 005

EMMANUEL CAINÃ ALVES DE MELO

SALAS DE AULA INTELIGENTES: UTILIZAÇÃO DE ARDUINO E BLUETOOTH LOW ENERGY COMO BEACONS PARA O MAPEAMENTO DE SALAS DE AULA

Monografia apresentada no curso de Sistemas de Informação da Universidade Federal do Ceará, como requisito parcial à obtenção do título de bacharel em Sistemas de Informação. Área de concentração: Computação.

Aprovada em:

BANCA EXAMINADORA

Prof. Dr. Marcio Espíndola Freire Maia (Orientador)
Campus Quixadá
Universidade Federal do Ceará – UFC

Prof. Me. Francisco Helder Candido dos Santos Filho
Campus Quixadá
Universidade Federal do Ceará - UFC

Prof. Dr. Paulo Antonio Leal Rego
Campus Quixadá
Universidade Federal do Ceará - UFC

A Deus, pela força e esperança que encontrei nele para trilhar meu caminho. A minha mãe e irmã, que nunca mediram esforços para que eu chegasse até aqui. A minha família, pelo apoio e compreensão e aos meus amigos, pelas alegrias e tristezas compartilhadas e por sempre me apoiarem e ajudarem nas horas difíceis.

AGRADECIMENTOS

Agradeço a Deus por todas as oportunidades que tive na vida.

Agradeço a minha mãe por todos os esforços, amor e carinho dedicados a mim.

Agradeço a minha família pela compreensão as minhas ausências.

Agradeço a minha namorada (e futura esposa) pelo companheirismo e dedicação doados a mim.

Agradeço ao Professor David Sena, não só por acreditar em mim, mas por fazer com que eu acreditasse em mim.

Agradeço ao meu Professor Orientador Marcio Maia pela ajuda e dedicação.

Agradeço a Regiane Ferreira e família por ter me acolhido.

Agradeço demais a todos os meus amigos. Agradeço por cada minuto ao lado de vocês. Conviver com vocês fez minha vida muito mais especial.

“ Tente uma, duas, três vezes e se possível tente a quarta, a quinta e quantas vezes for necessário. Só não desista nas primeiras tentativas, a persistência é amiga da conquista. Se você quer chegar a onde a maioria não chega, faça o que a maioria não faz.”

(Bill Gates)

RESUMO

Este trabalho propõe o desenvolvimento de uma plataforma para mapeamento de salas de aula através da tecnologia do Bluetooth Low Energy (BLE), com o objetivo de fornecer informações sobre as salas de aula, como: número da sala, número do bloco, aula atual, docente, disciplina, campus, instituição, horário de início e fim da aula, aula anterior e próxima aula, que serão mostrados aos usuários através de uma aplicação móvel. Trazendo dessa forma, o conceito de Internet das Coisas (IoT) para as salas de aula e facilitando o trabalho de alunos e professores, principal público alvo deste projeto.

Palavras-chave: Bluetooth Low Energy. Internet das Coisas. Aplicações Móveis. Salas de Aula Inteligentes.

ABSTRACT

This work proposes the development of a platform for mapping classrooms using Bluetooth Low Energy technology (BLE), with the objective of providing information about the classrooms, such as: room number, block number, current class, teacher, discipline, campus , Institution, start and end times of the class, previous class and next class, which will be shown to the users through a mobile application. Bringing this way, the Internet of Things (IoT) concept to the classrooms and facilitating the work of students and Teachers, the main target for this project.

Keywords: Bluetooth Low Energy. Internet of Things. Mobile Applications. Smart Classrooms

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE CÓDIGOS-FONTE

LISTA DE ABREVIATURAS E SIGLAS

IoT	Internet das Coisas / *Internet of Things*
BLE	*Bluetooth Low Energy* / *Bluetooth* 4.0
BPAN	*BlueTooth Personal Area Network*
RFID	*Radio-Frequency IDentification*
NFC	*Near Field Communication*
IP	*Internet Protocol*
IPv6	*Internet Protocol version 6*
6LoWPAN	*IPv6 over Low power Wireless Personal Area Networks*
GPS	*Global Positioning System*
Mbps	Megabit por segundo
SO	Sistema Operacional
IOS	*Iphone Operating System*
IDE	*Integrated Development Environment*
TI	Tecnologia da Informação
MQTT	*Message Queue Telemetry Transport*
M2M	*Machine-to-Machine* / Máquina a Máquina
JSON	*(JavaScript Object Notatio*
UUID	*Universally unique identifier*
PWM	*Pulse Width Modulation*

SUMÁRIO

1 INTRODUÇÃO

Pode-se entender o termo Internet das Coisas, ou *Internet of Things* (IoT), como ficou conhecido, como sendo um conjunto de tecnologias que engloba e conecta tudo e qualquer objeto no mundo físico através de uma representação virtual desses objetos e da utilização de um identificador único que os representa (LEE et al., 2015). Também pode ser entendido, como um fenômeno ou habilidade, em que vários objetos do cotidiano, como automóveis, geladeiras, fogões entre outros, se conectam através da internet e trocam informações entre si, monitoram e reagem à ações humanas (EVANGELATOS; SAMARASINGHE; ROLIM, 2012).

Ambas as definições reforçam a ideia de objetos tecnologicamente avançados que podem se conectar a internet e tem a habilidade de trocarem dados entre si, reagindo a determinadas ações humanas. Esta ideia de IoT, transmitida pelos autores Lee et al. (2015) e Evangelatos, Samarasinghe e Rolim (2012) será utilizada durante todo este trabalho.

O conceito de Internet das Coisas, somado a algumas tecnologias, pode ser utilizado para auxiliar empresas, consumidores, professores, alunos, empregados e a população de modo geral na realização de suas atividades e em suas vidas cotidianas. A dificuldade na obtenção de informações pelos alunos sobre determinados locais, como salas ou laboratórios de uma universidade, ou ainda constantemente atualizar as informações referentes a estes locais é um problema que pode ser resolvido com IoT, onde a "sala" comunicaria essas informações aos alunos. Um mundo conectado e acessível, onde as informações são obtidas quase que instantaneamente, é algo que vem sendo cada vez mais real e disponível através da Internet das Coisas.

Dentro da área de conhecimento da IoT, temos os prédios inteligentes, ou *Smart Buildings* que podem ser definidos como construções que fornecem um gerenciamento dos serviços de consumo de energia, gerência de medidas anti-desastres e promovem soluções para aumentar o conforto de seus ocupantes e a eficiência das tarefas realizadas (AGARWAL; WENG, 2012).

Uma das principais razões por trás da construção de prédios inteligentes e pela popularização do conceito, é o conforto decorrente do alto nível de automação que os prédios inteligentes provêm. Poder controlar a temperatura da casa de qualquer local, refrigerando-a ou aquecendo-a para que a temperatura dentro da casa esteja perfeita, antes mesmo de se chegar na casa, ou ainda, receber avisos da geladeira no *smartphone* toda vez que um alimento acabar ou estiver perto do fim são alguns exemplos da automação proveniente da utilização do conceito de

IoT.

Em uma construção inteligente é possível encontrar diversos tipos de componentes que empregam o conceito de IoT, como a criação de uma rede de comunicação *machine-to-machine* entre dispositivos de automação, torneiras que ajustam a temperatura da água de acordo com a temperatura ambiente e sensores que coletam e enviam informações sobre o local. Esses sensores podem ser utilizados para coletar informações, tais como: número de pessoas, temperatura atual, taxa de luminosidade, nível de CO_2 no ambiente, e muitos outros (ZAFARI; PAPAPANAGIOTOU; CHRISTIDIS, 2015). Alguns outros exemplos de componentes que utilizam-se do conceito de IoT para criação de prédios inteligentes são mostrados no projeto *IoT Home of the Future* da IEEE [1].

Em outras palavras, esses componentes possibilitam a construção ter um "cérebro" (SNOONIAN, 2003), permitindo-lhe tomar decisões, como aumentar ou baixar a temperatura de uma sala com base no número de ocupantes no local ou mostrar uma determinada informação sobre um conteúdo de uma aula para um aluno com base na sua proximidade da sala.

É sobre esse contexto, de Internet das Coisas e Prédios inteligentes que esse projeto trabalha, com o **objetivo geral** de desenvolver uma plataforma para mapear salas de aula e fornecer informações sobre as mesmas, como: identificação da sala, aula atual, professor atual, horário de inicio e fim da aula, instituição e próxima aula e aula anterior, visando a implantação da IoT como ferramenta para facilitar o trabalho de professores e alunos, principal público alvo deste projeto.

Além da seção de introdução, este trabalho é estruturado da seguinte maneira: **2** - Fundamentação Teórica, que mostra os conceitos chaves em que este trabalho se baseia, como IoT, *BlueTooth* de Baixa Potência, *Beacon* e MQTT; **3** - Objetivos, onde se é exibido o objetivo geral e objetivos específicos deste trabalho; **4** - Trabalhos Relacionados, que lista algumas abordagens utilizadas para implantação da IoT, através do *BlueTooth* de Baixa Potência e *Beacon*; **5** - Proposta, onde se é mostrado a arquitetura adotada na solução para o problema que este trabalho se dispõe solucionar, bem como os módulos que compõem o sistema; **6** - Apresentação e análise dos resultados, onde se é analisado os resultados gerados durante o projeto e também onde ocorre a validação desses resultados; e por fim a seção **7** - Considerações finais, onde se é feita as considerações finais e diretrizes futuras para este trabalho.

[1] Para mais informações acessar: http://transmitter.ieee.org/iot/rooms

2 FUNDAMENTAÇÃO TEÓRICA

Esta seção apresenta alguns conceitos básicos sobre o trabalho que devem ficar bem claros para que se possa compreender o contexto ao qual o projeto se aplica e a razão pela qual certas decisões foram tomadas. Primeiramente é explicado o principal conceito chave do trabalho, Internet das Coisas, que é o principal pilar onde este trabalho se sustenta, posteriormente se é falado sobre *Bluetooth* de Baixa frequência, uma tecnologia que está em ascensão e que é uma das maneiras mais recentes de se aplicar o conceito de IoT, depois é abordado o conceito de Beacon BLE, que são dispositivos cujo o objetivo é transmitir uma mensagem em *broadcast* para dispositivos em curto/médio alcance através da utilização do *Bluetooth* de Baixa frequência, e por fim, é falado sobre o protocolo de comunicação MQTT, bastante utilizado em aplicações IoT.

2.1 Internet das Coisas

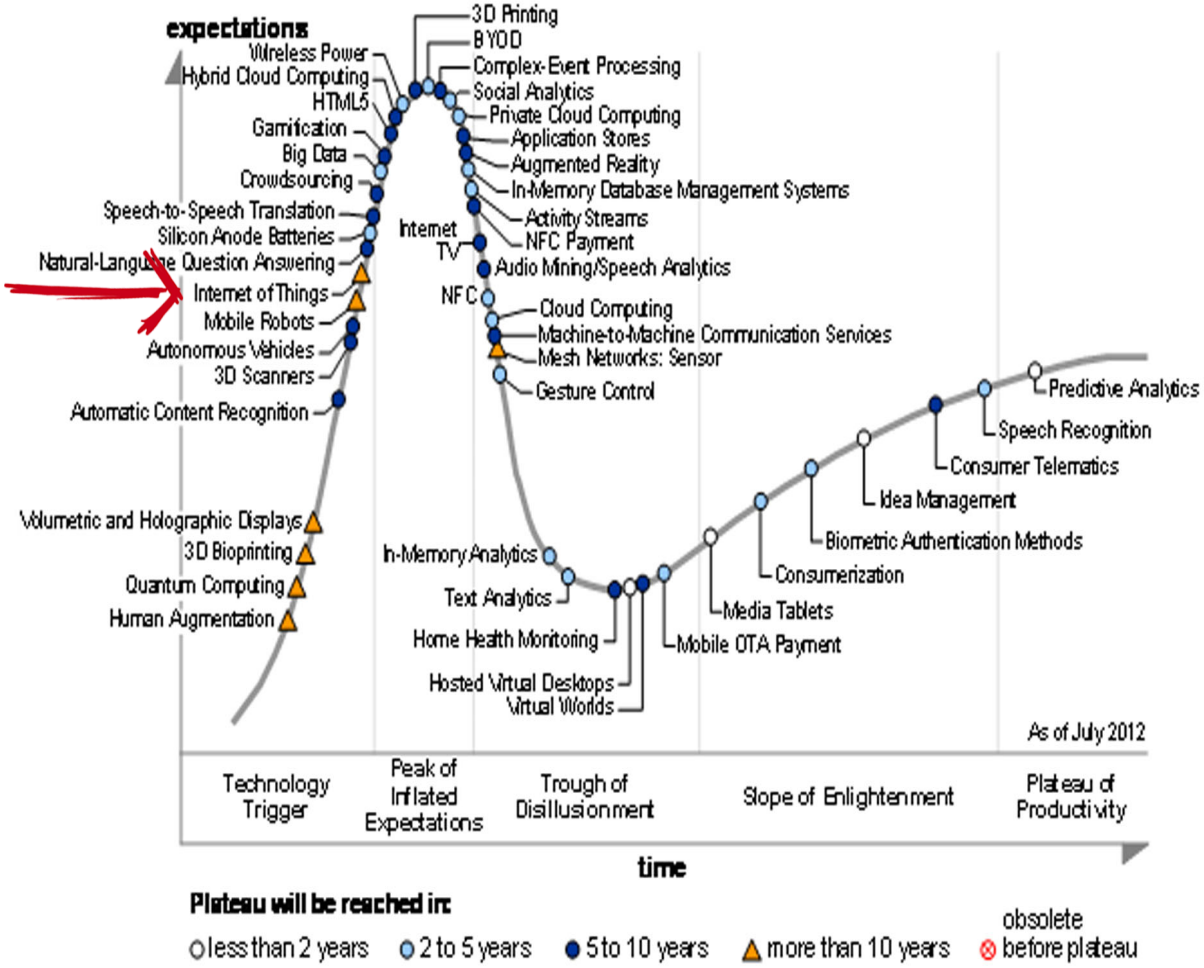
O termo Internet das Coisas é relativamente novo e foi formalizado pela primeira vez por Kevin Ashton no ano de 1999 (ASHTON, 2009), pesquisador e criador do sistema global para sensores *Radio Frequency Identification* (RFID), que é uma tecnologia de rádio de curto alcance usada para mapear e transmitir informações digitais, na sua maioria, através de dispositivos conhecidos como *TAGs* Inteligentes (LANDT, 2005) e (KORTUEM et al., 2010). Kevin Ashton no contexto de gestão de cadeia de abastecimento, definiu o termo IoT como obtenção de informações através de sensores computacionais (ASHTON, 2009). Apesar do pouco tempo de sua definição, o termo vem ganhando uma crescente popularização por enfatizar a representação virtual do mundo físico e das coisas que o compõe para permitir automação de aplicações que auxiliem as pessoas em suas tarefas (TAKALO-MATTILA; KILJANDER; SOININEN, 2013). Porém, boa parte do sucesso do termo também se deve, aos vários desafios enfrentados e pelas árduas conquistas adquiridas por seu precursor RFID, passadas tempos antes, abrindo as portas para o conceito de automação (WANT, 2006).

A perspectiva de Internet das Coisas como ferramenta que auxilia no dia-a-dia das pessoas, contribui para que o termo se torne cada vez mais popular. Council (2008) inclui IoT como uma das seis tecnologias que mais crescem na área da Tecnologia da Informação (TI) e previu que até o ano de 2025, a Internet das Coisas estará presente em tudo, incluindo em mobílias, eletrodomésticos e até mesmo embalagens de alimentos.

Pettey e Meulen (2012), classificaram a IoT no ano de 2012, como sendo uma das

mais promissoras tecnologias emergentes da área da Tecnologia da Informação (TI), prevendo uma ascensão no mercado do conceito e das tecnologias que o implementam entre 5 a 10 anos, com base nas pesquisas realizadas pela Gartner's Hype Cycle (ver figura 1) (GARTNER, 2012).

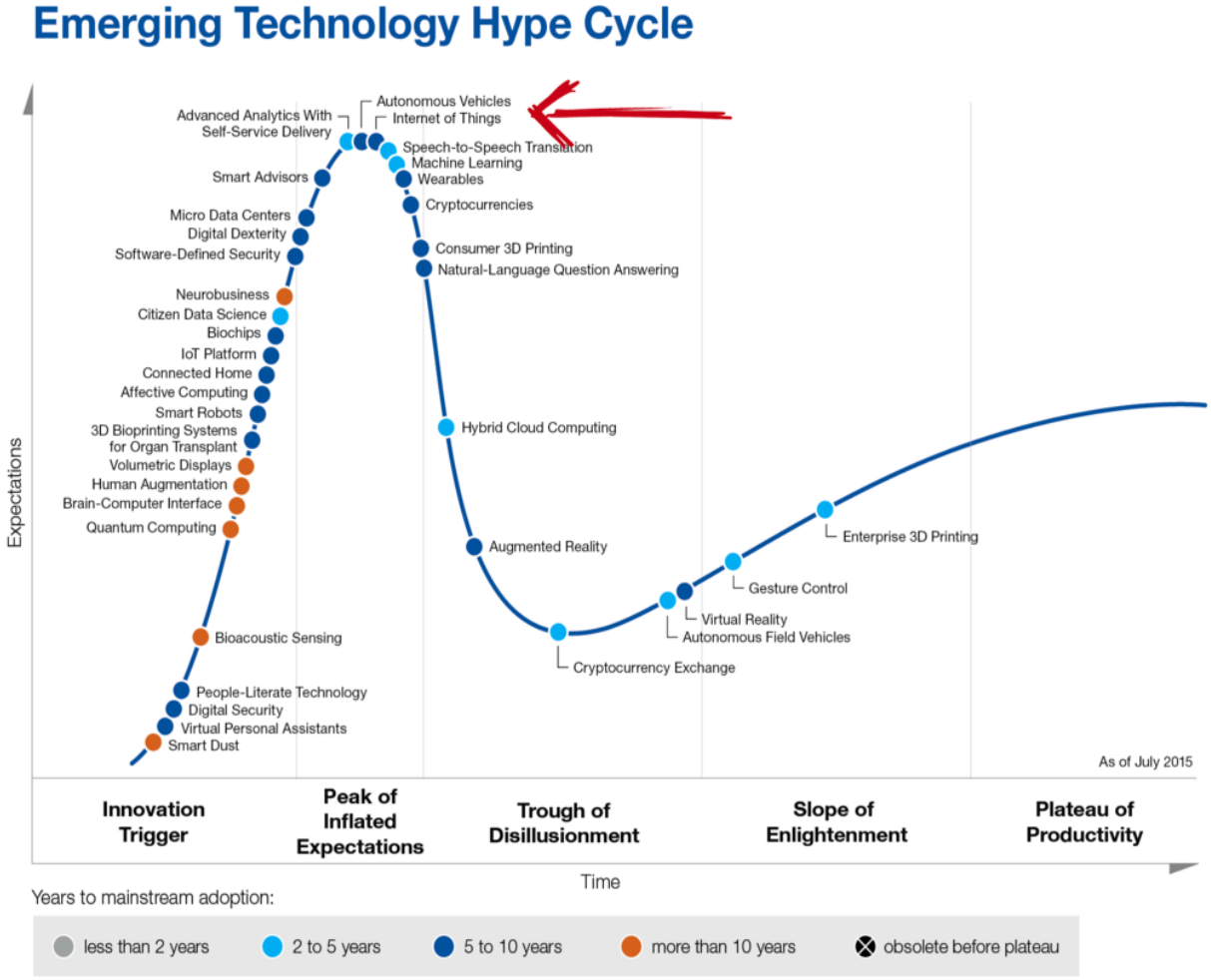
Figura 1 – Gartner 2012, Hype Cycle de tecnologias emergentes



Fonte: (PETTEY; MEULEN, 2012)

Hype Cycle é uma pesquisa que leva em consideração adoção, maturidade, impacto nas aplicações da atualidade e surgimento de tecnologias especificas. A pesquisa ainda nos mostra uma elevação no pico de expectativas inflacionadas (*peak of inflated expectations*), como podemos ver na figura 2, confirmando a previsão de Pettey e Meulen (2012).

Figura 2 – Gartner 2015, Hype Cycle de tecnologias emergentes



Fonte: (RIVERA; MEULEN, 2015)

O interesse e adoção de novos paradigmas, conceitos e tecnologias muda ao longo dos anos. Estes paradigmas, conceitos e tecnologias também crescem conforme o interesse por eles aumenta. A ferramenta *Google Trends* [1] da empresa Google, que mostra o interesse por determinados termos em um intervalo de tempo, com base no número de pesquisas sobre o termo na ferramenta de busca da empresa, externa o interesse pelo termo *Internet of Things* no ano de 2004 a 2016, como mostra a figura 3.

[1] https://www.google.com.br/trends/

Figura 3 – Pesquisa de interesse de 2004 a 2016 do termo Internet of Things

Fonte: adaptada pelo autor de Google

Como podemos ver na figura 3, é notável o crescente aumento de interesse pelo termo Internet of Things ao longo dos anos, chegando ao seu pico no ano de 2016. Isso implica que cada vez mais pessoas estão se interessando pelo conceito de IoT, o que leva a mais pesquisas e desenvolvimento de novas tecnologias que empreguem o conceito, gerando ainda mais popularidade do termo.

Na visão da IoT, basicamente qualquer coisa pode ser representada através do mundo virtual, desde os mais simples objetos, como uma lâmpada, um fogão ou uma porta, aos objetos mais complexos, como por exemplo: carros, prédios e televisões. Estes objetos, virtualizados, são definidos como "*Smart Objects*" (KORTUEM et al., 2010). Já no ano de 2011, o número de dispositivos conectados à internet ultrapassou o número de pessoas na Terra (GUBBI et al., 2013) e estima-se que esse número só venha a crescer. Zafari, Papapanagiotou e Christidis (2015) preveem que o número de dispositivos conectados à internet chegará a 24 bilhões no ano de 2020, o que se alinha com a previsão de Council (2008).

Conforme aumenta a variedade e quantidade de *Smart Objects*, são definidas áreas especificas para certos tipos de aplicações que utilizam IoT. De acordo com Gluhak et al. (2011), estas áreas podem ser classificadas conforme impacto causado, tipo de disponibilidade de rede, escala, cobertura, repetibilidade e envolvimento do usuário. O autor também define quatro áreas principais para a implantação do IoT:

1. **Pessoal e residencial** - aplicações que utilizam sensores que monitoram o corpo e saúde como os batimentos cardíacos, quantidade de passos etc, ou as aplicações que cuidam dos

serviços de casa, como controle da temperatura do ar condicionado, máquina de lavar, tempo de preparo dos alimentos no fogão, entre outros;

2. **Empresarial** - consiste na conexão das coisas dentro do ambiente de trabalho, visando melhora no desempenho dos funcionários e economia para a empresa. Emprega o conceito de Ambientes Inteligentes, gerindo a segurança do local, controlando o número de pessoas no ambiente, por exemplo, além de gerir os serviços públicos realizados no ambiente, como energia utilizada pelo ar-condicionado e as lâmpadas.
3. **Utilidades** - engloba os serviço de economia de custos e otimização de serviços, geralmente utilizados por empresas, para balancear a relação fornecimento/consumo de seus serviços. Como por exemplo, os sensores que desligam as luzes do local quando todos saem, ou ainda medidores de consumo de eletricidade inteligentes que marcam a hora do dia em que mais se consome energia;
4. **Móveis** - refere-se as aplicações que estão presentes em *Smartphones*, carros, sistemas de navegação etc, como localização pelo sistema de GPS (Global Positioning System) e velocidade média de um objeto conforme a velocidade de deslocamento no GPS; gerenciamento de sistema de entrega inteligente, que analisa dados sobre navegação, congestionamentos e melhor rota; e também inclui aplicações de monitoramento de produtos em tempo real.

Estas áreas estão fortemente interligadas, e constantemente trocam dados e serviços entre elas. Por exemplo, a área **Pessoal e residencial** produz dados que informam a quantidade de serviços elétricos utilizados e o custo dos mesmos, e torna esses dados disponíveis para uma empresa elétrica, no setor de **Utilidades**, que pode transformar esses dados em informações úteis para melhorar o serviço elétrico de uma cidade (GLUHAK et al., 2011).

2.2 Bluetooth de Baixa Potência

Bluetooth Low Energy (BLE), *Bluetooth* de Baixa Potência, *Bluetooth* 4.0 ou ainda *Smart Bluetooth* é um tipo de tecnologia de conexão sem fio desenvolvida pelo Grupo de Interesse Especial sobre *Bluetooth* (*Bluetooth Special Interest Group*) e pode ser entendida como uma especificação, ou funcionalidade que integra o *Bluetooth* desde 2010, a partir de sua versão 4.0 (BLUETOOTH, 2010b).

Bluetooth Low Energy possui algumas vantagens comparado ao *Bluetooth* Clássico, vantagens estas que o tornam mais "inteligente" ao proporcionar mais funcionalidades para

os desenvolvedores, fabricantes e para os usuários que terão uma melhor experiência com IoT fornecida pelo BLE. Segundo Bluetooth (2010a), essas vantagens fazem do BLE uma das tecnologias mais recomendadas para se empregar a Internet das Coisas, como as listadas abaixo:

- Promove a IoT através do suporte a diversas opções de conexão, como: *Internet Protocol version 6* (IPv6), 6LoWPAN (*IPv6 over Low power Wireless Personal Area Networks*) e *Bluetooth Smart Gateways*;
- Baixo consumo de energia, sendo capaz de funcionar por quase um ano com uma célula de bateria do tamanho de uma moeda;
- Capacidade de múltiplas conexões, onde temos um BLE central, chamado de BLE *master* que é responsável por aceitar, gerir e controlar as conexões de vários outros dispositivos que irão requisitar conexão, e são conhecidos como BLE *slaves*.
- Envio de arquivos para diversos dispositivos ao mesmo tempo, graças a capacidade de só permitir o envio de pequenas mensagens de no máximo 1 Mbps (Megabite por segundo), o BLE consegue enviar essas mensagens relativamente pequenas em *broadcast* para todos os BLE *slaves* conectados.
- Capacidade de enviar pequenas quantidade de bytes em *broadcast* sem a necessidade de criar uma conexão.
- Fornecer informações de sensores dos dispositivos que o utilizam, como: porcentagem da bateria, heixo de rotação, indicador de força de sinal recebido ou RSSI (*Received Signal Strength Indicator*), através do serviço de *response* sobre a mensagem em *broadcast* enviada.
- Baixo custo, encontrando no mercado, produtos com a tecnologia por menos de $10,00 dólares.

O *Bluetooth* de Baixa Potência vem sendo muito usado em sistemas de localização *indoor*. A *Apple*, empresa Norte-Americana que projeta e desenvolve produtos eletrônicos, utilizou a tecnologia do BLE para lançar o *Ibeacon*, um dispositivo capaz de localizar dispositivos IOS (*Iphone Operating System*) próximos que possuam a tecnologia BLE e enviar-lhes informações com base na localização desses dispositivos (CONTE et al., 2014).

Para o futuro, é previsto que o BLE esteja presente em mais de 1 bilhão de dispositivos devido ao crescente uso da tecnologia do *Bluetooth* Clássico, utilizada em celulares, carros, notebooks, aparelhos de som, dentre vários outros tipos de dispositivos. Esta popularização do *Bluetooth* Clássico serve como uma espécie de alavanca para impulsionar o

crescimento da tecnologia do BLE (GOMEZ; OLLER; PARADELLS, 2012). A 6LoWPAN, que também é o nome do grupo de desenvolvimento da *Internet Engineering Task Force* (IETF) e mantém o projeto que permite utilizar o IPv6 nas redes IEEE 802.15.4. Hui e Culler (2008), reconheceu a importância do *Bluetooth Low Energy* como tecnologia emergente para aplicação da Internet das Coisas.

A Capacidade de receber conexões de diversos aparelhos, enviar uma quantidade de bytes sem precisar abrir conexões e a capacidade de enviar pequenos arquivos de uma vez para vários dispositivos que estejam conectados, chamaram a atenção para o *Bluetooth Low Energy*. Entretanto, a característica que mais despertou interesse, foi a capacidade de enviar dados sem a necessidade de estabelecer uma conexão, um recurso essencial para este trabalho. Estes mecanismos encontrados na tecnologia do BLE, mostram-se uma proposta interessante para as instituições que aderirem ao projeto e tornam a experiência nas salas de aula (campo de aplicação deste projeto), algo bem mais prático e automático.

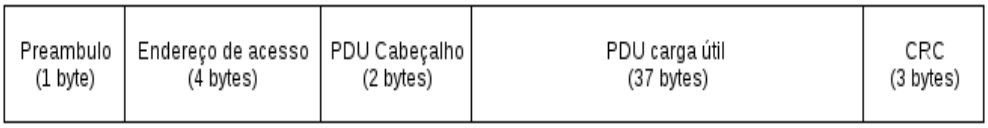
2.3 Beacons BLE

Um *Beacon* BLE pode ser entendido como um dispositivo que utiliza da Tecnologia do *Bluetooth low Energy* para transmitir uma mensagem em *broadcast* para todos os dispositivos que estejam em um raio de cerca de 100 metros e possuam a função BLE ativada (ZAFARI; PAPAPANAGIOTOU, 2015). Os dispositivos, geralmente *smartphones* e *tablets*, reagem a essa mensagem realizando alguma ação referente a mensagem enviada.

É importante frisar que *Beacons* são dispositivos periféricos de comunicação e não devem ser utilizados para trabalhar com grandes volumes de dados e sim, apenas transmitir pequenas mensagens contendo a informação desejada. Como disse Patrick Leddy, chefe executivo e fundador da empresa *Pulsate - mobile marketing firm*, os *Beacons* funcionam apenas como uma espécie de farol (por isso a nomenclatura “Beacon”, do português, “faról”), apenas enviando mensagens com uma pequena quantidade de dados em *broadcast* para os dispositivos ao redor, os informando sobre sua existência (STATLER, 2016).

Na figura 4, podemos ver como uma mensagem enviada por um *Beacon BLE* é estruturada:

Figura 4 – Estrutura do pacote de uma mensagem BLE

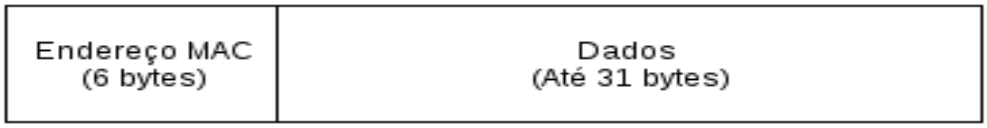


Fonte: adaptada pelo autor de Bluetooth (2010a)

O pacote de uma mensagem BLE pode variar de 8 a 47 bytes de tamanho, contendo os seguinte atributos: **preâmbulo:** utilizado para gerenciar os protocolos internos. As mensagens em *broadcast*, por exemplo, possuem preâmbulo igual a 10101010; **endereço de acesso**: sempre será 0x8E89BED6 para as mensagens enviadas em *broadcast*; **PDU cabeçalho:** que contem informações sobre os dados que serão enviados, como o tipo e o tamanho; **PDU carga útil:** que contem os dados propriamente dito, com a informação desejada; e por último, **CRC**: calculado em ralação ao PDU cabeçalho mais PDU carga útil (BLUETOOTH, 2010a).

Dentro do PDU carga útil, ainda temos o endereço MAC (*Media Access Control*, do português, Controle de Acesso de Mídia), que contém o identificador único do dispositivo que está mandando a mensagem, e como podemos ver na figura 5, sobram 31 bytes para serem utilizados (BLUETOOTH, 2010a).

Figura 5 – Estrutura do atributo PDU carga útil de uma mensagem BLE



Fonte: adaptada pelo autor de Bluetooth (2010a)

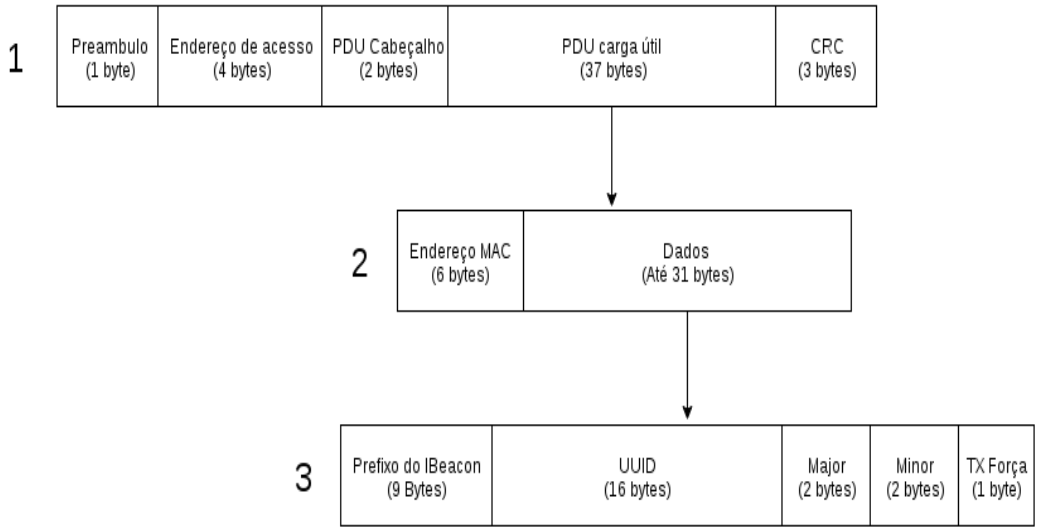
Os autores Zafari, Papapanagiotou e Christidis (2015) também descrevem os *Beacons* como ferramentas utilizadas principalmente com o objetivo de identificar *smart objects* próximos que estejam com a função do BLE ativada e mostrar conteúdos específicos para aquele dispositivo com base na sua localização e proximidade de determinado *Beacon*. Em um de seus trabalhos, (LEE et al., 2015) utiliza um *BLE Beacon* para localização *indoor* utilizando um filtro de passagem baixo para aumentar a precisão da localização, que se trata de uma localização mais precisa de objetos em interiores.

O conceito de *Beacon* ficou mais conhecido por um produto lançado pela gigante *Apple*, como citado na seção **2.2**: o *IBeacon*. *IBeacon* é um protocolo específico baseado na tecnologia do BLE e se destina e ser usado como um *BLE Beacon*, porém com enfoque nos dispositivos com o sistema *IOS (Iphone OS)*, utilizado pelos *smartphones* da empresa (ZAFARI;

PAPAPANAGIOTOU, 2015).

Na figura 6, podemos ver como o protocolo IBeacon está estruturado, dentro do pacote de mensagem BLE.

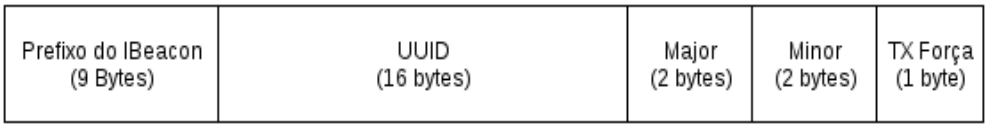
Figura 6 – Protocolo IBeacon dentro da estrutura do pacote da mensagem BLE



Fonte: adaptada pelo autor de Bluetooth (2010a)

O protocolo IBeacon não deve ultrapassar o tamanho máximo de 31 bytes, que é o tamanho de dados livres dentro do PDU carga útil. O protocolo ocupa 30 bytes e contem os seguintes atributos, como podemos ver na figura 7: **prefixo do *Beacon***, que contém informações avançadas sobre o *Beacon*, como por exemplo se o modulo BLE do *Beacon* aceita conexões, a identificação do fabricante do *Beacon*, o tipo do *Beacon* e um cabeçalho avançado; o **UUID** (Universally Unique IDentifier, do português, Identificador Único Universal) que serve para identificar o *Beacon* unicamente em relação a ouros *Beacons*; **Major:** que serve para identificar um conjunto de *Beacons* em uma área maior, como os *Beacons* de uma cidade x; **Minor**: que serve para identificar um conjunto específico de *Beacons*; e **TX força**, que é utilizado para medir a distância do *Beacon* com base na força do sinal que ele está emitindo.

Figura 7 – Protocolo IBeacon



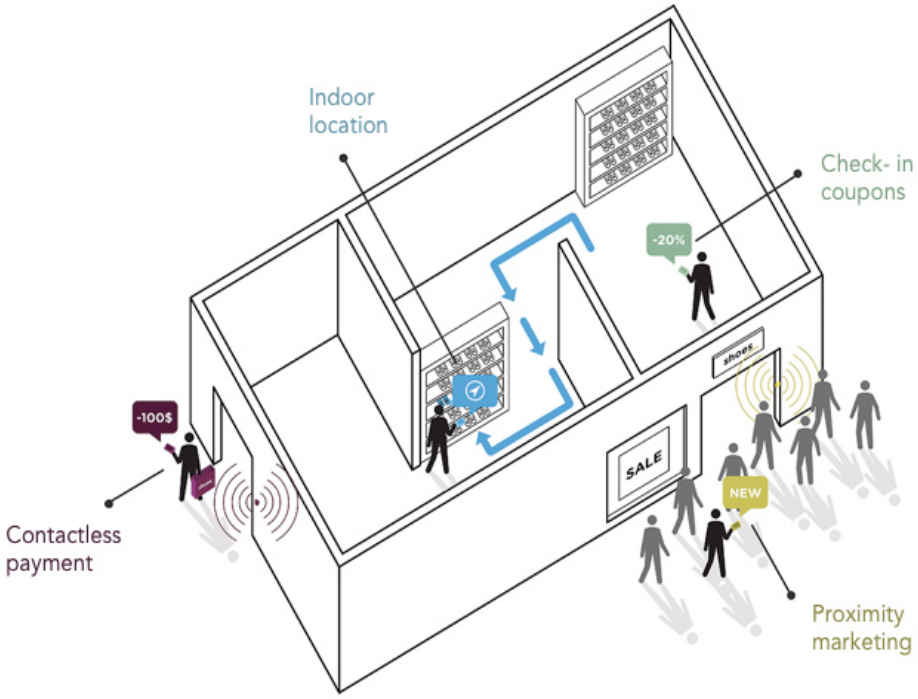
Fonte: adaptada pelo autor de Bluetooth (2010a)

Além dos *Beacons* serem usados como ferramentas para calcular a localização de determinada aplicação e ser utilizado em sistemas de localização *indoor* de alta precisão, sua

principal utilização é na transmissão de dados para dispositivos próximos, sendo bastante úteis para transmitir informações específicas para os dispositivos com base na sua localização atual. Por exemplo, o *Beacon* pode mostrar uma informação sobre um determinado produto de uma loja a um cliente que se aproxima de determinado *Beacon*, ou mostrar descontos quando se passa em frente a um estabelecimento, ou ainda informar qual aula está sendo ministrados em determinada sala, quando um aluno se aproxima do local.

A Figura **8** exemplifica uma aplicação dos *Beacons* em uma loja, onde os cliente entram em uma região denominada *Beacon's Zone* e recebem informações sobre os produtos, cupons de descontos personalizados, informações sobre o caminho mais fácil para chegar a seção onde determinado produto está localizado e novidades sobre a loja.

Figura 8 – Exemplo de aplicação dos *Beacons* em uma loja.



Fonte: Gigaom (2016)

Com a capacidade de se obter a localização de dispositivos próximos aos *Beacons* e a transmissão de dados em *broadcast* para todos os dispositivos em um raio de 100 metros, os *Beacons* se mostram bastante interessante para o contexto desse trabalho e por isso foi utilizado na criação do mesmo.

2.4 MQTT

MQTT ou Fila de Mensagem para Transporte de Telemetria (*Message Queue Telemetry Transport*) é um protocolo de comunicação extremamente leve e rápido, desenvolvido originalmente pela empresa IBM[2] e é utilizado principalmente na comunicação M2M (*machine-to-machine* ou máquina a máquina) (LAMPKIN et al., 2012).

Além de ser extremamente rápido e simples, o MQTT trás diversos benefícios para dispositivos com limitações de recursos, como por exemplo os dispositivos embarcados (MQTT, 2016). Alguns desses benefícios são:

- Assíncrono: ou seja, não depende de resposta de outros dispositivos, ideal para casos em que a conexão com a internet não é confiável;
- Mensagens curtas: o cabeçalho do pacote enviado com o protocolo MQTT é extremamente simples, deixando o pacote basicamente com os dados desejados, o que é essencial em casos de largura de banda baixa;
- Recursos limitados: não requer muito processamento nem memória do cliente que o implemente, fazendo o protocolo compatível com quase todos os tipos de dispositivos;
- Comunicação otimizada: fornece opções bastante otimizadas para sensores e dispositivos remotos
- Escalabilidade: é bastante escalável, através da utilização do esquema de tópicos, o que facilita muito a manutenção das soluções que utilizam o protocolo.

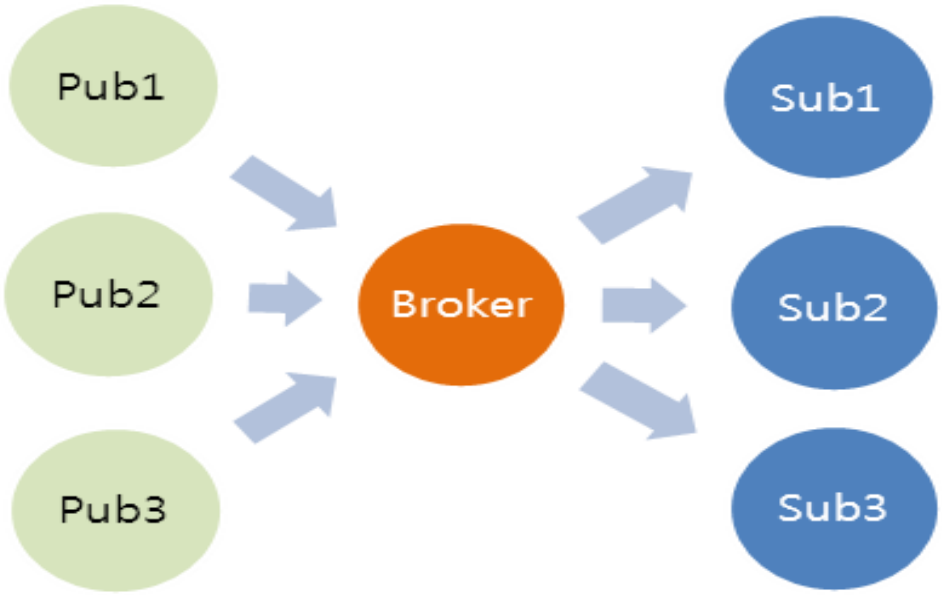
Essas e outras características, fazem do MQTT uma das principais escolhas quando se pensa em comunicação M2M, principalmente quando se trata da comunicação entre dispositivos embarcados e sensores, tornando o protocolo ideal na implantação do conceito de IoT.

O protocolo MQTT implementa a arquitetura *publish/subscriber* (publicadores/assinantes) na sua comunicação, com um servidor central conhecido como *broker* (intermediário), além disso, o protocolo gira em torno do conceito de tópicos, onde todos os clientes se inscrevem, tanto para enviarem informações, como para recebe-las (LAMPKIN et al., 2012). Projetado para ser fácil de utilizar e ter milhares de clientes conectados em um mesmo *broker* (de acordo com a capacidade do servidor), os utilizadores do MQTT, afirmam que a arquitetura *publish/subscriber* implementado pelo protocolo, é o que é preciso para se construir uma rede de comunicações IoT (COLLINA; CORAZZA; VANELLI-CORALLI, 2012).

[2] http://www.ibm.com/br-pt/

De acordo com a arquitetura *publish/subscriber*, uma comunicação através do protocolo MQTT, funciona da seguinte maneira: o *publisher*, ou "escritor", envia/publica as informações para o *broker* (servidor) através de um tópico, os *subscribers*, ou inscritos nesse mesmo tópico, recebem essas informações do *broker*, e por fim, o *broker* gerencia e repassa essas informações, ou seja, todo o trabalho referente a gerência da informação enviada, fica a cargo do *broker*, como mostrado na figura 9. É importante notar, que apesar de ser uma comunicação M2M, um dispositivo (cliente) não se comunica diretamente com o outro, não há comunicação ponto-a-ponto ou P2P (*Peer-to-Peer*).

Figura 9 – Arquitetura do protocolo MQTT



Fonte: (XAPPSOFTWARE, 2014)

Tanto *publishers* como *subscribers*, são clientes, na arquitetura implementada pelo MQTT, e conectam-se a apenas um *broker*, porém, um mesmo cliente MQTT, pode ser *subscriber* e *publisher* de diversos tópicos (LAMPKIN et al., 2012).

Uma mensagem no protocolo MQTT é dividida em duas partes: **tópico** e ***payload***. Um tópico é uma *string* (similar a uma *url*) e é utilizada para implantar uma hierarquia entre os tópicos, o que facilita a capacidade de um cliente publicar e se inscrever em um tópico especifico, ou mesmo em todos os tópicos de uma subárea. Na *string* de um tópico, se utiliza o caractere barra inclinada a direita ("/") para separar os diversos níveis do tópico, como por exemplo: area/area_id/sensor/sensor_id/. Por fim, temos a ultima parte do pacote, o *payload*, onde se encontra a informação, propriamente dita (HUNKELER; TRUONG; STANFORD-CLARK, 2008).

3 OBJETIVOS

A seguir são descritos o **objetivo geral** e **objetivos específicos** deste trabalho.

3.1 Objetivo Geral

O **objetivo geral** deste trabalho é desenvolver uma plataforma para mapear salas de aula, a qual irá fornecer informações sobre as mesmas, tais como: identificação da sala, através do número da sala e bloco onde se encontra; nome da disciplina da aula atual; nome do docente que está ministrando a aula atual; horário de início e fim da aula, instituição; próxima aula e aula anterior, visando a implantação da IoT como ferramenta para facilitar o trabalho de professores e alunos.

3.2 Objetivos específicos

Os seguintes **objetivos específicos** devem ser atingidos para que se possa concluir o trabalho:

- Criar um *Beacon* utilizando uma placa de prototipação para programação de software embarcado;
- Desenvolver um aplicativo *mobile* que irá receber as informações enviados pelo *Beacon* e apresentá-las ao usuário;
- Implementar um servidor web que será utilizado pelos docentes para atualizar as informações transmitidas pelo *Beacon*;
- Implantar e testar o projeto em um ambiente real de execução.

4 TRABALHOS RELACIONADOS

Como dito na seção **1** de introdução deste trabalho, o conceito de IoT pode ser entendido como um conjunto de tecnologias que engloba e conecta tudo e qualquer objeto no mundo físico, provendo uma troca de informações destes objetos (LEE et al., 2015), porém uma das principais dificuldades encontradas por quem quer empregar o conceito de IoT, seja em empresas, salas de aula, construções ou áreas de convivência é prover essa conexão, ou ainda, realizá-la de forma independente provendo o máximo possível de automação sem que para isso o usuário deva interagir para que ocorra o funcionamento do sistema.

A proposta de solução deste problema varia muito conforme a escolha das tecnologias que se pretende trabalhar. Khoo (2015), por exemplo, propõe uma solução para automação de conexões mobile para interação com um dispositivo de vídeo através de um servidor, que por sua vez irá gerenciar estas conexões. O autor aponta em seu trabalho alguns benefícios gerados pela utilização da IoT, porém também mostra algumas dificuldades enfrentadas pelo seu uso, como garantir conexões através de um identificador único para um grande número de dispositivos conectados a internet. Com base nesse problema, o autor realiza um estudo de algumas tecnologias e lista os benefícios e limitações das mesmas. Depois o autor escolhe as tecnologias disponíveis que melhor se adaptam ao seu contexto.

Khoo (2015) escolhe a ferramenta *IBeacon* da *Apple*, equipada com a tecnologia BLE (*BlueTooth Low Energy*), assim como a tecnologia BPAN (*BlueTooth Personal Area Network*), que se trata de uma outra especificação do *BlueTooth*, como o BLE. A solução do autor para o problema encontrado consiste em utilizar a tecnologia BPAN para prover acesso à um servidor através de uma rede de conexões IP (*Internet Protocol* ou Protocolo de internet), sobre o *BlueTooth*. O Autor também estrutura sua plataforma criada com essas tecnologias no modelo Cliente-Servidor, onde uma placa *Raspberry Pi* faz a função de servidor enquanto uma aplicação mobile desenvolvida em *Android* faz o papel de cliente.

O autor não deixa claro onde será implantado seu projeto, porém informa que a plataforma deve ser capaz de informar aos usuários sobre sua própria existência e deve fornecer uma conexão de forma automática e instantânea aos seus usuários. Através dos resultados obtidos e da análise dos mesmos, Khoo (2015) deixa claro a capacidade da plataforma desenvolvida por ele e a necessidade das ferramentas escolhidas. Os resultados obtidos pelo autor ajudaram nesse trabalho na escolha das ferramentas trabalhadas pelo autor e pelo modelo da estrutura do projeto desenvolvido por Khoo (2015).

O seguinte projeto utiliza um *Beacon* que implementa o protocolo *IBeacon*, assim como Khoo (2015). No inicio do projeto, a abordagem de Khoo (2015) foi testada, porém viu-se que não era viável criar uma conexão com os dispositivos moveis dos clientes, tanto pela demora da conexão, tendo em vista que alguns clientes podem apenas passar em frente uma sala e querer ver de imediato as informações referente a ela, quanto pelo número de clientes (dispositivos móveis) que o projeto proposto pode atender de uma só vez, o que necessitaria de um equipamento mais robusto. Sendo assim, a solução ajudou na escolha das ferramentas e na arquitetura do projeto, como dito antes, porém sua solução em si não é viável para este projeto.

Em Conte et al. (2014) os autores propõe uma solução para o problema de detecção de ocupantes de uma construção utilizando a plataforma que eles chamaram de *BLUE-SENTINEL* para localizar e quantificar os ocupantes em construções e utilizar estas informações para aprimorar as decisões tomadas pelos *smart buildings*, com o principal fim de economizar energia destes locais (desligando serviços de cômodos desocupados) e aumentar o nível de conforto dos usuários nas construções (com base nas informações de quando alguém entra ou sai e quem é essa pessoa).

A plataforma denominada de *BLUE-SENTINEL* consiste na implementação do protocolo *IBeacon* da *Apple* em um *Beacon* customizado, utilizando uma placa Arduino para criá-lo. Conte et al. (2014) utiliza esse *Beacon* customizado para mapear os locais de um prédio e disparar eventos nas aplicações dos *smartphones* dos usuários, que por sua vez irão enviar para um servidor chamado de *BLUE-SENTINEL Core* e que é responsável por monitorar e quantificar todos os dispositivos dos usuários presentes na construção, informações sobre a distância entre o *Beacon* e o aparelho e também sobre quem é o usuário. O servidor por sua vez, irá repassar essas informações para o Sistema Gerenciador da Construção (Building Management System) que irá tomar a melhor decisão do que fazer com base nessas informações.

Assim como em (CONTE et al., 2014) nós também iremos criar um *Beacon* utilizando um modulo BLE e uma placa Arduino para tal, bem como utilizá-lo para mapear determinados locais, no caso salas de aula. Porém também usaremos esse *Beacon* para transmitir informações através de seu identificador para as aplicações dos usuários e não apenas como meio para mapeamento de locais.

Em Takalo-Mattila, Kiljander e Soininen (2013) é mostrado uma abordagem para representar semanticamente objetos físicos reais utilizando *Smart Bluetooth Beacons* para mapear esses objetos e enviar um identificador único chamado de *ucode* para a aplicação do usuário. A

aplicação por sua vez, requer a resolução do ucode por um servidor, após recuperar a informação, a aplicação retorna a aplicação móvel do usuário, o endereço no banco de dados referente ao objeto de interesse. Por fim, a aplicação do usuário utiliza este endereço para consultar o banco de dados e obter as informações desejadas sobre o determinado objeto.

Na abordagem de Takalo-Mattila, Kiljander e Soininen (2013) também é criado uma aplicação *mobile* para dispositivos *IOS* que escaneia todos os *Smart Bluetooth Beacons* vizinhos ao dispositivo com o objetivo de mapear e mostrar informações sobre esses objetos representados através dos *Smart Bluetooth Beacons*. Assim como na proposta de Takalo-Mattila, Kiljander e Soininen (2013) usaremos os *beacons* desenvolvidos para representar algo do mundo físico, no caso as salas de aula, como também criaremos uma aplicação *mobile*, porém a aplicação será para dispositivos *Android*, com o objetivo de apresentar as informações referentes a sala, entretanto estas informações estarão presentes tanto no servidor, como na aplicação, através de um banco local, solucionando o problema de falta de conexão com a internet.

5 PROPOSTA

Para a criação do sistema proposto, de mapeamento de salas de aula, vários fatores tiveram que ser analisados antes de se iniciar o processo de desenvolvimento, como: identificação de modelos atuais de mapeamento utilizando BLE; problemas e soluções com esses modelos, como visto na seção **4** de trabalhos relacionados; escolha de um modelo e escolha das tecnologias e ferramentas que foram utilizadas nesse projeto.

A análise desses fatores ocorreu na etapa de pesquisa desse projeto, que teve fundamental importância para tomada de algumas decisões no projeto, como a arquitetura do sistema e os módulos que a compõe e as interfaces de comunicação entre estes módulos, como veremos a seguir.

5.1 Arquitetura do Sistema

O sistema proposto neste trabalho consiste em três módulos, como podemos ver na figura 10: um *Web Service* desenvolvido utilizando NodeJS (*JavaScript*), que também possui uma interface de usuário, que por sua vez foi desenvolvida utilizando AngularJS[1] e serve para armazenar e repassar as informações relacionadas às salas de aula para os *Beacons* específicos; uma plataforma de prototipagem Arduino UNO com um módulo *Bluetooh Low Energy* e um *Ethernet Shield*, que recebe através da rede as informações que são envidas pelo *Web Service*; e por fim, um módulo para dispositivos móveis Android, que é uma aplicação para a plataforma Android, utilizada para mostrar as informações das salas para os usuários.

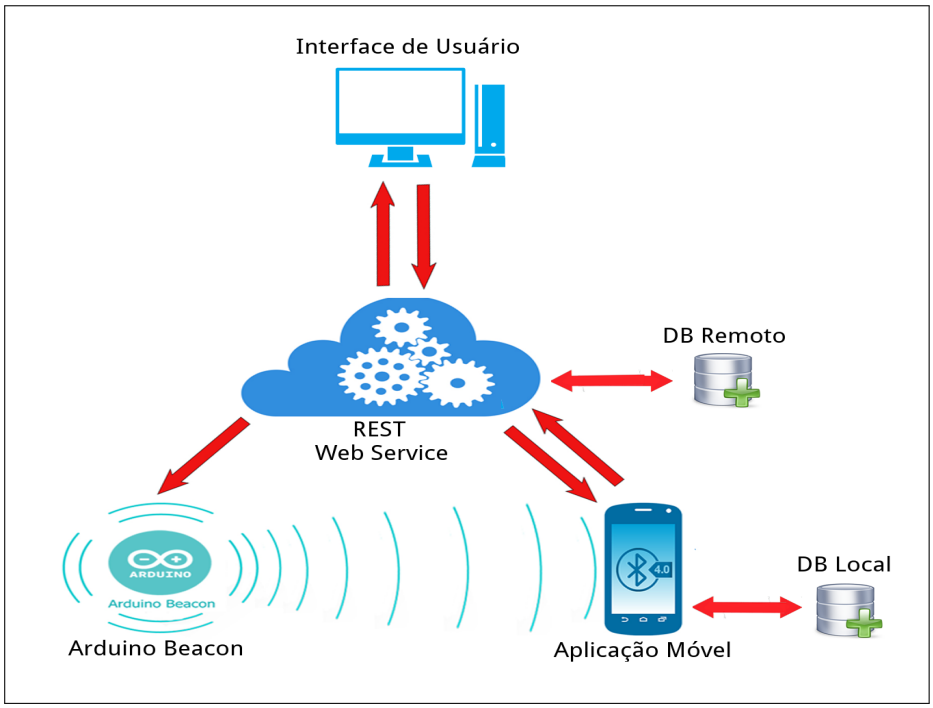
A arquitetura escolhida é baseada na arquitetura cliente-servidor, onde se tem dois principais clientes, o Arduino Beacon e a Aplicação Móvel. O servidor web utiliza o MongoDB para armazenar as informações cadastradas através da interface de usuário, essas informações serão recuperadas posteriormente para serem enviadas para o Arduino Beacon, que por sua vez irá repassar a referencia dessas informações para a aplicação móvel mostrá-las ao usuário. Esta comunicação entre os módulos do sistema será posteriormente explicada com mais detalhes.

Um dos problemas encontradas nos modelos pesquisados é a dependência da conexão *com Web Server* que a aplicação *mobile* possui, tornando impossível mostrar alguma informação ao usuário, caso não haja essa conexão. Pensando em solucionar esse problema, criamos um banco local para a aplicação, que é utilizado quando não se tem conexão com o servidor e pode

[1] https://angularjs.org/

ser atualizado posteriormente, quando se tiver conexão.

Figura 10 – Arquitetura do Sistema



Fonte: Desenvolvida pelo Autor.

A seguir, são descritos cada um desses módulos mais detalhadamente e as tecnologias utilizadas para desenvolve-los.

5.1.1 Módulo Web Service

O módulo ***Web Service*** é responsável por cadastrar e armazenar as informações sobre as salas de aula, que serão transmitidas pelo Arduino Beacon. Como dito antes, o servidor web foi desenvolvido em JavaScript, através do *framework* NodeJS[2]. O servidor foi construido com base na arquitetura REST (*REpresentational State Transfer*) ou em tradução livre para português, Transferência de Estado Representacional, que utiliza o protocolo HTTP de comunicação, para prover acesso e possibilitar modificações aos recursos através dos métodos *GET, PUT, DELETE* e *POST*, do protocolo HTTP (PEREIRA, 2014). A principal razão para a escolha dessas ferramentas foi a facilidade no desenvolvimento que o *framework* NodeJS proporciona e a liberdade que a arquitetura REST disponibiliza para representar os dados (ou recursos), podendo ser escolhidos formatos como xml, txt ou JSON. Para este projeto foi escolhido JSON

[2] https://nodejs.org/en/

(*JavaScript Object Notation*) para representar os dados, devido a sua popularidade.

As informações cadastradas no banco são salvas através do banco de dados MongoDB[3], que é um banco de dados não relacional, baseado em modelo de dados de documentos, bastante similares ao JSON (ORGANIZATION, 2016). O MongoDB é recomendado para todos os níveis de projetos, desde projetos complexos a simples aplicações WEB, devido as facilidades que a ferramenta traz, como consultas textuais, capacidade de dividir o banco de dados entre diversas máquinas e o fato de oferecer esquemas dinâmicos, o que torna o processo de mudanças estruturais muito mais simples (ORGANIZATION, 2016). Justamente por estas vantagens apresentadas pelo MongoDB, que este foi escolhido para este projeto.

No servidor deverão ser cadastradas as seguintes informações para serem transmitidas pelo Arduino Beacon: sala, bloco, disciplina, docente, hora de inicio, hora de fim, próxima aula, aula anterior, campus e instituição, que serão chamados de **informações sobre a sala**. Como dito na seção 2.3, deve-se evitar transmitir uma grande quantidade de dados pelo *Beacon*, pois como vimos, ainda na seção 2.3, o tamanho de dados disponíveis para enviar as informações desejadas é muito pequeno. A solução encontrada foi enviar para o *Beacon* apenas os identificadores dessas informações, fazendo com que a mensagem transmitida pelo *Beacon* respeite as restrições do protocolo de envio, além da mensagem ser transmitida, pelo *Beacon* quase que instantaneamente, devido seu pequeno tamanho.

Após os dados referente as informações sobre a sala serem cadastrados com sucesso, é criado um identificador único para cada atributo, chamado de “keyNome_do_atributo”, que será utilizado para criar uma mensagem com as *keys* de todos os atributos. A seguir segue o trecho do código em JavaScript referente ao processo descrito a cima.

```
historico.getLast(function(err, historic) {
    var keyHistoric = 0;
    if(historic && historic.length > 0 && historic[0])
        keyHistoric = historic[0].key;
    var message = "";
    message += intToHex(object.keySala, 2);
    message += intToHex(object.keyBloco, 2);
    message += intToHex(object.keyDisciplina, 2);
    // Hora de início
    message += intToHex(new Date(object.timestampBegin).getHours() + 1, 2);
    message += intToHex(object.keyDocente, 4);
    message += intToHex(object.keyInstuicao, 2);
    // Minuto de início
    message += intToHex(new Date(object.timestampBegin).getMinutes() + 1, 2);
```

[3] https://www.mongodb.com/

```
15      message += intToHex(object.keyCampus, 2);
        message += intToHex(object.keyDisciplinaPrevious, 2);
17      message += intToHex(object.keyDisciplinaNext, 2);
        // Hora de fim
19      message += intToHex(new Date(object.timestampEnd).getHours() + 1, 2);
        message += intToHex(object.keyHistorico, 2);
21      message += intToHex(1, 2);
        message += intToHex(1, 2);
23      // Minuto de fim
        message += intToHex(new Date(object.timestampEnd).getMinutes() + 1, 2);
25      // Retorna a mensagem mapeada com os identificadores das informações
        callback(message);
27  });
```

Código-fonte 1 – Criando a mensagem que será transmitida pelo Arduino Beacon

Após criada, a mensagem com as informações referentes à sala será enviada para o *Beacon* e posteriormente será transmitida por ele. Para tal, utilizamos o Mosca, um MQTT Broker para NodeJS, leve e rápido, para publicar em um tópico gerado dinamicamente com base nas informações cadastradas sobre o *Beacon*. A seguir segue o código em JavaScript referente a esta parte.

```
1   eventos.map(function(v) {
        Beacon.getByKey(v.keyBeacon, function(err, data) {
3           if(err || !data || data.length == 0) {
                console.log("Erro ao atualizar evento!");
5           } else {
                var beacon = data[0];
7               __getInstituicaoCampus(v.keyInstituicao, v.keyCampus, function (instituicao,
                    campus) {
                    Evento.generateMessage(v, function (message) {
9                       var topic = instituicao[0].name.toUpperCase() + "/" + campus[0].name.
                            toUpperCase() + "/" + beacon.name;
                        console.log("Compartilhando o evento [" + message
11                          + "] com o beacon " + beacon.name + " width " + topic);
                        __mosca.publish(topic, message);
13                  });
                })
15          }
        })
17  });
```

Código-fonte 2 – Publicando a informação em um tópico gerado dinamicamente

O *Web Service* também é responsável por atualizar o banco de dados local da aplicação móvel, que funciona através de um método *GET*, que é disparado por uma comparação entre a ultima data de atualização do *Web Service* (histórico) e a última data de atualização do banco local da aplicação.

5.1.2 Módulo Arduino Beacon

O módulo **Arduino Beacon** é encarregado de transmitir as informações sobre a sala de aula para as aplicações móveis, através do módulo BLE. O Arduino *Beacon* é formado por um módulo *Bluetooth* de Baixa Potência, de referência HM-10 *Master/Slave*, um *Shield Ethernet*, de referência HanRun HR911105A e a própria placa de prototipagem Arduino Uno.

Como dito antes, a plataforma de prototipagem escolhida foi o Arduino UNO, tanto pela sua simplicidade e facilidade de se trabalhar quanto pela sua popularidade, que torna a comunidade da placa muito rica em conteúdo e suporte. O módulo Ethernet foi escolhido pelas bibliotecas disponíveis para ele e pelo ótimo desempenho que a placa apresenta, demorando em alguns casos menos de 1 seg para estabelecer uma conexão. Já o módulo BLE foi escolhido principalmente pela versão de seu chipset, sendo na época da aquisição um dos mais modernos, por operar como um *Beacon*, através da utilização do protocolo *IBeacon* e pelo alcance de seu sinal. Na tabela 1, podemos ver algumas características do módulo *bluetooth* escolhido.

Tabela 1 – Especificações Bluetooth 4.0 HM-10 Master/Slave

Especificações BLE	
Chipset	CC2541
Versão	V4.0 BLE
Banda	2.4GHz ISM
Tensão de Operação	3.3/5V
Corrente de Funcionamento	50mA
Alcance	Até 100 metros

Fonte: Desenvolvida pelo Autor a partir do trabalho de Karydis (2015)

Antes de utilizar o módulo BLE como um *Beacon*, primeiramente tivemos que utilizar os comandos **AT** para configurá-lo, fazendo com que o módulo operasse como um IBeacon. A lista de comandos AT seguidos encontra-se no Código-fonte 3.

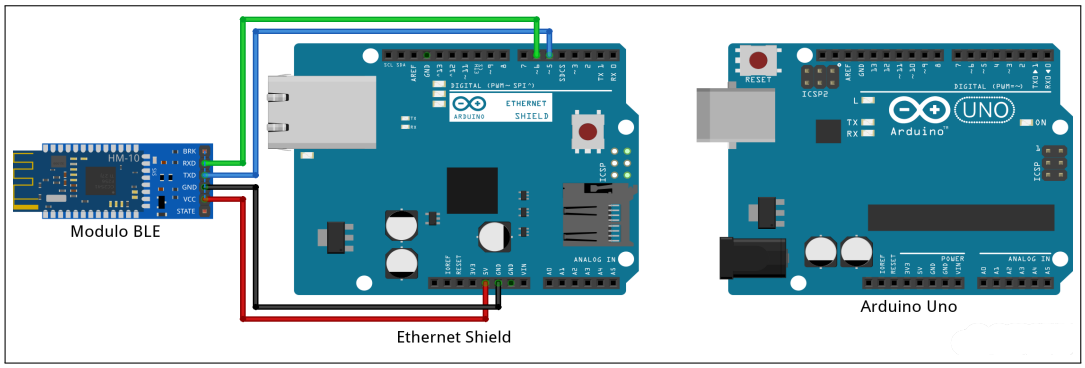
```
1 AT + RENEW          // Restaurar as configurações de fábrica.
  AT + RESET          // Reiniciar HM-10.
3 AT                  // Recebe uma mensagem de OK.
  AT + MARJ0x1234     // Colocar Major número para 0x1234 (hexadecimal).
```

```
5  AT + MINO0xFA01      // Colocar Minor número para 0xFA01 (hexadecimal).
   AT + ADVI5           // Colocar o intervale de envio de mensagem para 5 (546.25 milliseconds)
7  AT + NAMEQBeacon-0   // Colocar o nome do módulo. Tenha certeza que será único.
   AT + ADTY2           // Habilita conexões e RESPONSE.
9  AT + IBEA1           // Habilitar o modo IBeacon.
   AT + PWRM0           // Permite o auto-sono do módulo. Isso reduz o consume de energia de 8
       para 0.18 mA (Miliampere).
11 AT + RESET           // Reiniciar HM-10.
```

Código-fonte 3 – Comandos AT para configuração IBeacon

O *shield Ethernet* é encaixado sobre a placa Arduino Uno, estendendo suas portas de saída, porém o *shield Ethernet* utiliza os pinos 10, 11, 12 e 13 para o envio e recebimento de dados. Deve-se prestar bastante atenção nesse detalhe, pois para evitar conflitos entre informações não se deve ligar nada nessas portas. Sendo assim o modulo BLE foi ligado a outras saídas PWM (Pulse Width Modulation), no caso as saídas 5 e 6 do *shield Ethernet*, como mostrado a figura 11 desenvolvida utilizando a ferramenta *Fritzing*[4], que retrata a comunicação entre os módulos do Arduino para formar o Arduino Beacon.

Figura 11 – Circuito módulo BLE mais Ethernet Shield e Arduino UNO



Fonte: Desenvolvida pelo Autor.

5.1.3 Módulo Aplicação Móvel

O módulo da Aplicação Móvel, desenvolvido para *smartphones*, é responsável por mostrar ao usuário as informações enviadas pelo *Beacon*. Nesta parte do sistema os ID (Keys) enviados pelo *Beacon* são pesquisados no banco de dados através do *framework* sugarRecord[5]

[4] http://fritzing.org/home/
[5] http://satyan.github.io/sugar/

que utiliza o MySQL[6] para a persistência dos dados. A aplicação móvel foi desenvolvida para *smartphones* com o sistema operacional Android, tanto por ser o sistema operacional *móvel* mais utilizado, como pelo suporte e ajuda da comunidade.

Semelhante ao que ocorre no Servidor Web, a aplicação *mobile* percorre a mensagem enviada pelo *Beacon*, porém extraindo os IDs referentes as informações da sala, como podemos ver no Código-fonte 4. É importante lembrar que a mesma ordem seguida no servidor para construir a mensagem com os identificadores de cada campo, deve ser utilizada para extrair esses identificadores, caso o contrário podem ocorrer erros ao mostrar um valor que não corresponde ao campo, como por exemplo uma sala de número “UFC”.

```
1  //métodos para extrair as informações do pacote enviado pelo Beacon
   private MessageSector[] messageSectors = new MessageSector[] {
3      new MessageSector(2, Sala.class),
       new MessageSector(2, Bloco.class),
5      new MessageSector(2, Disciplina.class),
       // Hora início
7      new MessageSector(2, Integer.class),
       new MessageSector(4, Docente.class),
9      new MessageSector(2, Instituicao.class),
       // Minuto início
11     new MessageSector(2, Integer.class),
       // Aulas anterior e próxima
13     new MessageSector(2, Campus.class),
       new MessageSector(2, Disciplina.class),
15     new MessageSector(2, Disciplina.class),
       // Hora fim
17     new MessageSector(2, Integer.class),
       new MessageSector(2, Historico.class),
19     new MessageSector(2, Integer.class),
       new MessageSector(2, Integer.class),
21     // Minuto fim
       new MessageSector(2, Integer.class),
23 };
```

Código-fonte 4 – Método para extrair os IDs da mensagem enviada pelo *Beacon*

Após recuperar os dados referentes a cada identificador, a aplicação móvel mostra esses dados para o usuário. O método também verifica se algum campo está nulo, e caso esteja, é pedido ao usuário que atualize sua versão local do banco de dados.

```
1  public void run() {
       tvSala.setText(sala != null? "Sala " + sala.getName(): "...");
```

6 https://www.mysql.com/

```
3       tvBloco.setText(bloco != null? "Bloco " + bloco.getName(): "...");
        tvDocente.setText(docente != null? docente.getName(): "...");
5       tvDisciplina.setText(disciplina != null? disciplina.getName(): "...");
        tvInstituicao.setText(instituicao != null? instituicao.getName(): "...");
7       tvCampus.setText(campus != null? campus.getName(): "...");
        tvAulaAnterior.setText(aulaAnt != null? aulaAnt.getName(): "...");
9       tvAulaProxima.setText(aulaProx != null? aulaProx.getName(): "...");
        tvInicio.setText(String.format("%02d", horaI) + ":" + String.format("%02d", minI));
11      tvFim.setText(String.format("%02d", horaF) + ":" + String.format("%02d", minF));
        //Se alguns dos campos estiver nulo, significa que o banco local est\ a desatualizado
13      if(! dialogShowed && (sala == null || bloco == null || docente == null || disciplina ==
            null
                || instituicao == null || campus == null || aulaAnt == null
15              || aulaProx == null)) {
            dialogShowed = true;
17          //Pede para o usuário atualizar o banco local
            new MaterialDialog.Builder(BeaconActivity.this)
19                  .title("Dados desatualizados")
                    .content("Conecte com a internet e atualize seus dados!")
21                  .positiveText("OK")
                    .show();
23      }
    }
```

Código-fonte 5 – Método para mostrar as informações recuperadas do banco de dados

5.2 Interfaces

Como dito na seção 5.1.1, o Web Service cria os tópicos em tempo de execução de acordo com as informações cadastradas sobre os Beacons, como por exemplo, através dos campos **instituição**, **campus**, **sala** e **nome do Beacon** o web service consegue criar uma rota e publicar as mensagens destinadas aos *Beacons* daquele tópico. Isso deve ocorrer porque um *Broker* gerencia vários inscritos e em vários tópicos e é responsável por atualizar as informações dos tópicos a qual esses inscritos fazem parte. Entretanto, os inscritos só se inscrevem em um tópico por vez e levando isso para o campo de atuação deste trabalho, que são salas de aula, um *Beacon* instalado em uma sala, dificilmente será removido de lá, sendo assim,o tópico a qual o *Beacon* está escrito deve ser fixo.

No código-fonte 6, podemos ver um exemplo do Arduino Beacon se inscrevendo no tópico “UFC/QUIXADA/BEACON-0”

```
#include <SPI.h>
```

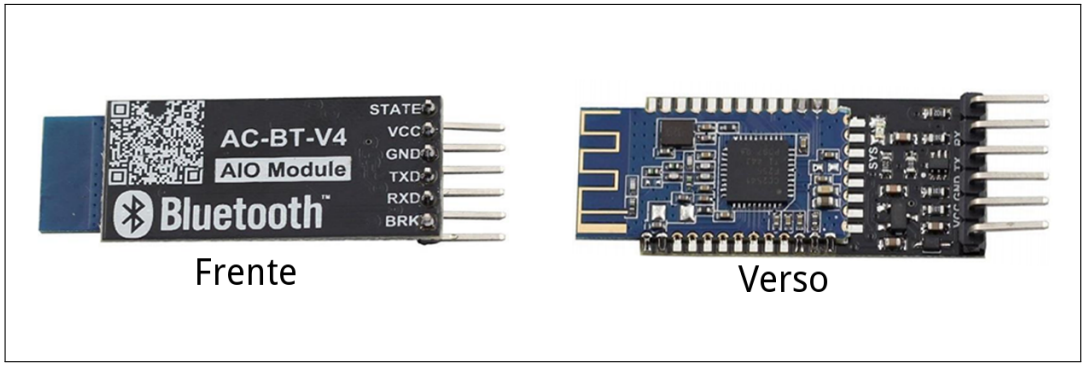
```
#include <Ethernet.h>
#include <PubSubClient.h>
// inicializando os atributos para o cliente ethernet.
byte mac[] = {  0xDE, 0xED, 0xBA, 0xFE, 0xFE, 0xED };
IPAddress ip(192, 168, 0, 156);
//IP BROKER
IPAddress server(192, 168, 0, 112);
//Criando o cliente ethernet.
EthernetClient ethClient;
PubSubClient client(ethClient);
void setup() {

  if(Ethernet.begin(mac) == 0) {
    Serial.println("Falha ao configurar Ethernet usando DHCP. Tentando ip arbitrario");
    Ethernet.begin(mac, ip);
  }
}
void loop(){
  /* Ethernet */
  if (!client.connected()) {
    reconnect();
  }
  updater.loop();
}

void reconnect() {
  // Repete ate conseguir uma conexao;
  while (!client.connected()) {
    if (client.connect("BEACON-0")) {
      //Tópico a qual o Beacon irá se inscrever
      client.subscribe("UFC/QUIXADA/BEACON-0");
    } else {
      Serial.print("Ethernet: Falha ao receber conexao!, ");
      Serial.print(client.state());
      delay(5000);
    }
  }
}
```

Código-fonte 6 – Beacon se subscrevendo em um tópico fixo

Após se inscrever, o *Beacon* começa a receber as mensagens que são publicadas naquele tópico. Depois de receber uma mensagem, que é um conjunto de IDs, o *Beacon* quebra essa mensagem em 4 novas mensagens e registra cada uma delas em uma faixa UUID (*Universally unique identifier*) do módulo BLE (esse processo será melhor explicado na seção

5.2.1 - protocolo). Depois de registrar o UUID, que agora contém todos os IDs enviados pelo servidor, o *Beacon* começa a enviar através do modulo BLE (figura 12) a mensagem em *broadcast* contendo o UUID com as informações referente aos dados da sala.

Figura 12 – Módulo Bluetooth 4.0 HM-10 Master/Slave



Fonte: Desenvolvida pelo Autor.

O módulo BLE envia o UUID para a aplicação mobile utilizando o protocolo IBeacon (ver figura 6). Na aplicação Mobile, a mensagem é capturada através da biblioteca AltBeacon[7], que é uma biblioteca que permite os dispositivos Android utilizarem os Beacons, semelhante aos dispositivos *IOS*, como mostra o Código-fonte 7.

```
beaconManager.addRangeNotifier(new RangeNotifier() {
    @Override
    public void didRangeBeaconsInRegion(Collection<Beacon> beacons, Region region) {
        if (beacons.size() > 0) {
            Beacon b = beacons.iterator().next();
            while (b != null) {
                onBeaconReceived(b);
                b = beacons.iterator().next();
            }
        }
    }
}
```

Código-fonte 7 – Recebendo uma mensagem de um *Beacon* através da Biblioteca AltBeacon.

[7] http://altbeacon.org/

5.2.1 Protocolo e Dados

A ideia por trás da adaptação do protocolo IBeacon que nós tivemos foi aproveitar os bytes que eram utilizados apenas como identificador para mapear as informações de uma sala de aula através de um ID, que tem um tamanho de 2 hexadecimais, para cada informação referente a sala. A tabela 2 nos mostra como é estruturado o UUID e os comandos **AT** necessários para modifica-los.

Tabela 2 – BLE Universally Unique Identifier (UUID)

UUID: 00000001-00000001-0000-0001-00000001			
AT+IBE000000001	AT+IBE100000001	AT+IBE200000001	AT+IBE300000001

Fonte: Desenvolvida pelo Autor.

Com a mensagem enviada pelo Web Server, que como dito antes é um conjunto de IDs, o Arduino *Beacon* quebra essa mensagem em 4 conjuntos de 8 hexadecimais (ou 4 bytes) e armazena cada um em uma faixa do UUID, como mostra o Código-fonte 8.

```
void callback(char* topic, byte* payload, unsigned int length) {
    String message = "";
    for (int i = 0; i < length; i++) {
        message += (char)payload[i];
    }
    Debug::i(String("Mensagem recebida [") + topic + "]: " + message);
    ble.setUUID(0, message.substring(0, 8));
    ble.setUUID(1, message.substring(8, 16));
    ble.setUUID(2, message.substring(16, 24));
    ble.setUUID(3, message.substring(24, 32));
}
```

Código-fonte 8 – Divisão das informações em faixas do UUID

Seguindo o padrão formado na hora do cadastro das informações no Web Service, e depois de ter dividido as informações nas 4 faixas do UUID, teremos o seguinte mapeamento das informações, como mostra a tabela 3.

Tabela 3 – Adaptação do protocolo IBeacon UUID

UUID				
	ID de sala	ID de bloco	ID de disciplina	Hora de início
AT+IBE0:	00	00	00	00
	ID de docente	ID de docente	ID de instituição	Minutos de início
AT+IBE1:	00	00	00	00
	ID de campus	ID de aula anterior	ID de próxima aula	Hora de fim
AT+IBE2:	00	00	00	00
	ID de histórico	ID livre	ID livre	Minuto de fim
AT+IBE2:	00	00	00	00

Fonte: Desenvolvida pelo Autor.

Como o UUID é utilizado para identificar os *Beacons* e agora está sendo utilizado para outro fim, utilizamos o nome do modulo BLE como identificador, criando uma restrição no servidor para que não seja possível cadastrar dois *Beacons* com o mesmo nome.

6 APRESENTAÇÃO E ANÁLISE DOS RESULTADOS

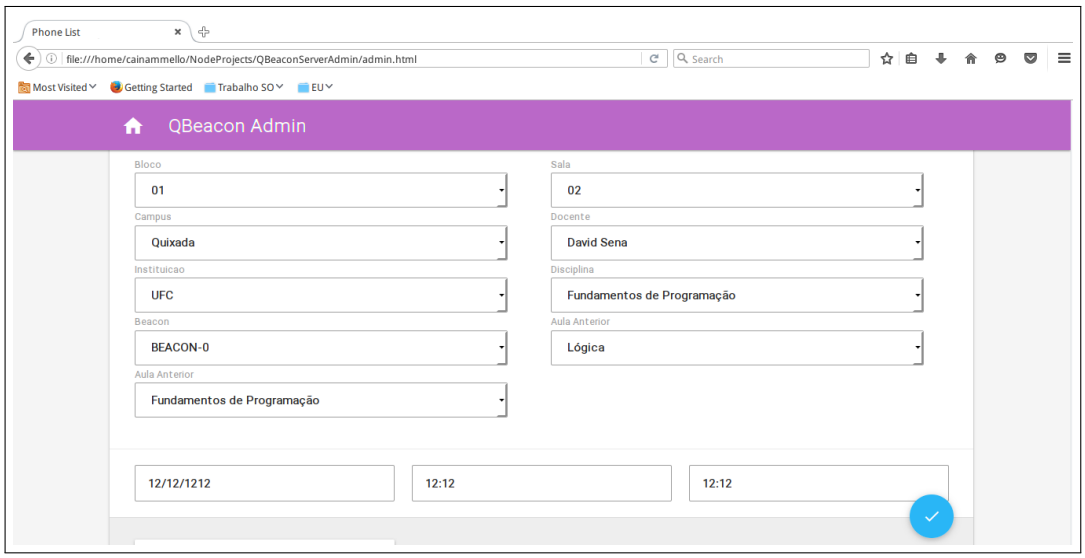
Nesta seção são apresentados os resultados do trabalho de implementação de uma plataforma para mapear salas de aula e fornecer informações sobre elas, além dos problemas encontrados durante todo o processo de pesquisa e desenvolvimento.

Trabalhar com novas tecnologias e ferramentas sempre é um desafio interessante. Apesar do BLE existir desde 2012, ainda é difícil achar algo concreto sobre a tecnologia. As várias definições, diversos modelos de equipamento, bibliotecas com restrições por modelos e falta de suporte foram os principais problemas enfrentados, definindo o escopo do trabalho em mapear e mostrar informações sobre as salas.

Entretanto, essas dificuldades fizeram com que o projeto amadurecesse bastante, tornando o objetivo geral mais claro e conciso. Durante o desenvolvimento do projeto, conseguimos cumprir nossos objetivos específicos, que eram a criação do *Beacon* utilizando o Arduino e BLE; Desenvolver uma aplicação mobile para apresentar os dados aos usuários; Criar o servidor, com um cliente *front-end*; e por fim, testar o projeto, porém não tivemos muito tempo para evoluir esta ultima parte.

A plataforma cumpriu seu objetivo de conseguir mostrar as informações sobre as salas, deixando a desejar apenas no número de *Beacons* que conseguimos montar, no caso 01, devido a falta de equipamento e tempo para comprar novas peças. Entretanto ainda conseguimos empregar os conceitos propostos pelo projeto. A figura a 13 mostra a tela de cadastro da aplicação Web, para adicionar as informações sobre uma sala.

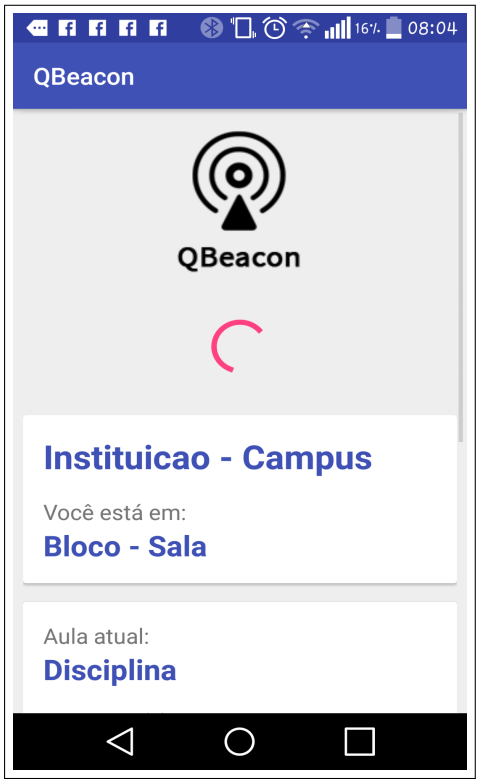
Figura 13 – Tela de Cadastro da aplicação Web



Fonte: Desenvolvida pelo Autor.

Após ser cadastrada, as informações sobre a sala serão publicadas via MQTT em um tópico que é formado pela junção dos campos: instituicao/campus/beacon/. Assim que a mensagem for publicada o *Beacon* já será atualizado e começará a transmitir as informações para os dispositivos móveis próximos ao *Beacon*. A figura 14 mostra a tela de início do aplicativo, buscando por *Beacons* na proximidade.

Figura 14 – Tela de início da aplicação móvel



Fonte: Desenvolvida pelo Autor.

Em seguida, a aplicação móvel recebe a mensagem enviada pelo *Beacon* e atualiza as informações na tela, como mostra a figura 15.

Figura 15 – Campos atualizados após o recebimento da mensagem pelo Beacon

Fonte: Desenvolvida pelo Autor.

Apesar de ser apenas um protótipo, a aplicação se comportou de maneira bastante satisfatória, sendo possível executá-la sem maiores problemas.

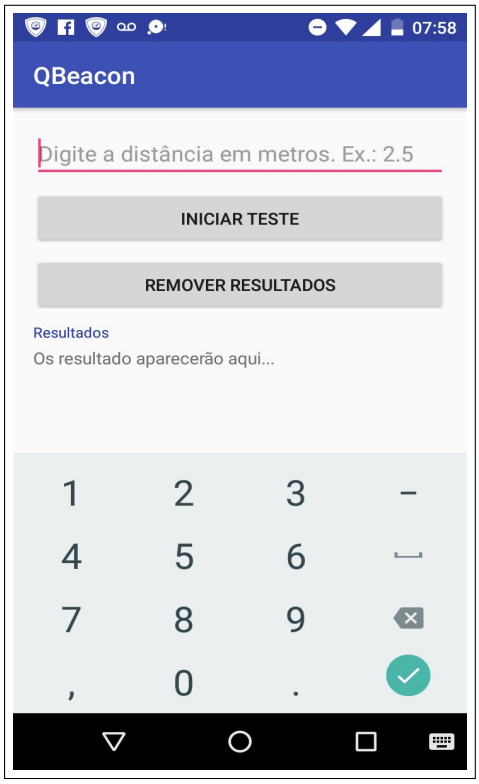
6.1 Avaliação

Para avaliar a plataforma, um teste de desempenho foi executado com o objetivo de avaliar o tempo que a aplicação, logo após sua inicialização, demora para encontrar um *Beacon*, variando a sua distância. Também foi medido o tempo para atualizar as informações na tela com base nos dados enviados pelo *Beacon*.

O teste foi realizado com 4 aparelhos diferentes. Todas as aplicações em *background* foram encerradas e o Wi-Fi desligado. Os métodos utilizados para realizar este teste foram os Códigos-fonte 5 e 7, ambos da seção 5.1.

Para tornar os testes mais práticos e rápidos, foi criada uma aplicação baseada nos dois métodos citados a cima, como mostra a figura 16. O teste basicamente consiste em entrar com um valor no campo de *input* de texto referente a distância em metros do *Beacon* e analisar o tempo de resposta das funções de detecção dos *Beacons* e atualização dos dados na tela.

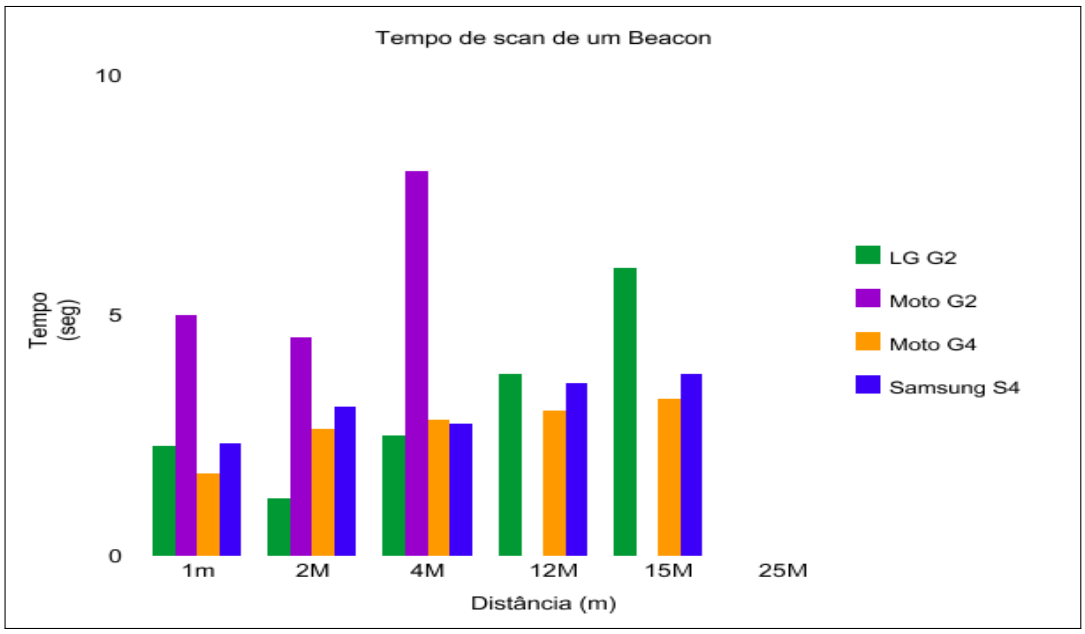
Figura 16 – Aplicação utilizada no teste



Fonte: Desenvolvida pelo Autor.

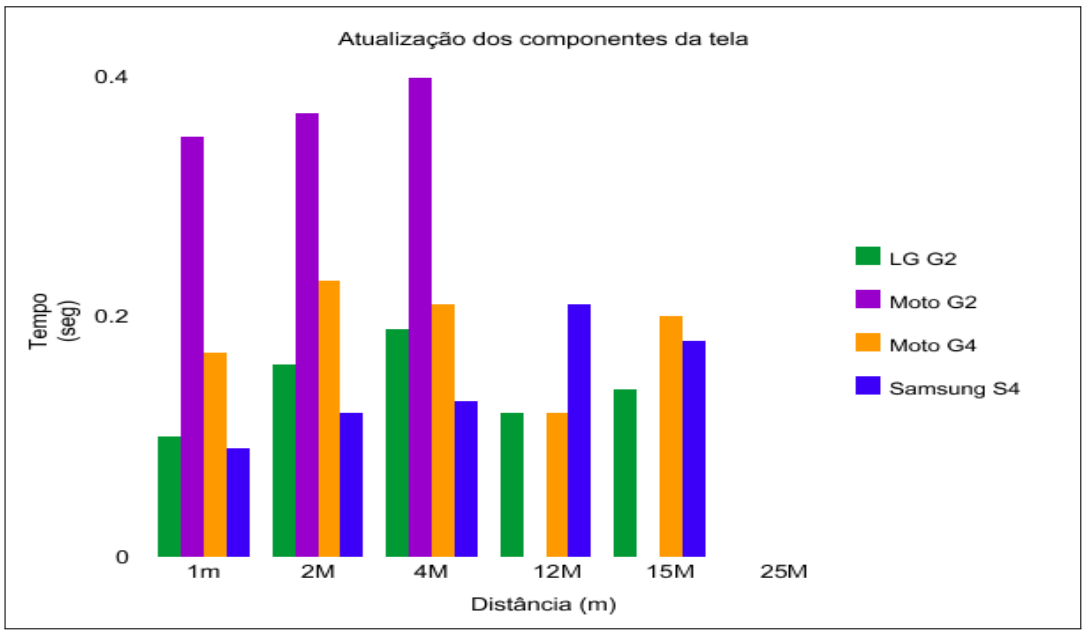
Os testes foram executados cerca de 3 vezes para cada dispositivo nas distâncias de 1,2,4,12,15 e 25 metros, escolhemos o menor valor das 3 execuções para comparar os dispositivos. A figura 17 representa a relação entre o tempo para detectar um *Beacon* e a distância desse *Beacon*. Já a figura 18, representa a relação entre o tempo para atualizar os dados na tela e a distância do *Beacon*.

Figura 17 – Tempo para reconhecer um *Beacon* x distância



Fonte: Desenvolvida pelo Autor.

Figura 18 – Tempo para atualizar os dados na tela x distância



Fonte: Desenvolvida pelo Autor.

Como podemos observar na figura 17, a distância é um forte indicador no tempo de reconhecimento dos *Beacons*. Essa diferencia é bem visível analisando os dados gerados pelo Moto G2. Outro fato curioso é que mesmo o módulo *Bluetooth Low Energy* HM-10, utilizado nesse experimento, especificar que seu sinal pode ser enviado a quase 100 metros, a partir de 20

metros nenhum dispositivo o reconheceu mais.

Analisando a figura 18, podemos perceber algumas alterações no tempo para visualizar uma informação, mas nada que seja significativo, pois alguns dispositivos como o Sansung S4 e o LG G2, não gastam mais de 0.2 segundos para realizar essas tarefa. Sendo assim, podemos concluir que na maioria das vezes que a aplicação está demorando em mostrar os dados para o usuário na tela, se da devido ao fato do *Beacon* ainda não ter sido encontrado e não por algum atraso na interface gráfica.

7 CONSIDERAÇÕES FINAIS

Com este projeto, podemos observar o quão ainda podemos explorar tecnologias como o *Bluetooth Low Energy*, que surgiram há relativamente pouco tempo, mas já colecionam uma grande contribuição no desenvolvimento e difusão do conceito de Internet das Coisas.

Ao longo do desenvolvimento deste projeto, surgiram algumas dúvidas, como se era viável mandar grandes quantidades de dados sobre o BLE, ou se simplesmente ele foi feito só para representar um local físico através de um identificador. Entretanto uma simples análise de outra perspectiva nos mostrou que ainda temos muito o que aprender com essas tecnologia e que ela pode sim se tornar o futura da IoT em um futuro não tão distante.

Concluímos aqui este trabalho, fazendo uma recapitulação dos objetivos específicos alcançados, como: criação do *Beacon* utilizando o Arduino e BLE; Desenvolver uma aplicação mobile para apresentar os dados aos usuários; Criar o servidor, com um cliente *front-end*; e por fim, testar o projeto. E através destes objetivos específicos, chegamos ao nosso objetivo geral que era construir uma plataforma para o mapeamento de salas de aula, fornecendo informações sobre ela como: número da sala, bloco, aula atual, docente, horário de inicio e fim da aula, instituição, aula anterior e próxima aula.

Tudo na computação pode melhorar, e nosso projeto não é diferente. Segue algumas melhorias que podem ser aplicadas ao projeto em trabalhos futuros:

- Acrescentar mais funcionalidades à aplicação *android*, como o tempo que o aluno ficou na sala de aula, com base no número de mensagens que ele recebeu do *Beacon* daquela sala em um intervalo de tempo;
- Propor um meio de tornar mais genérico a forma com que o mapeamento dos bytes no UUID são feitos
- Otimizar o *response* do BLE para conseguir mandar mais dados sem deixar o processo de comunicação lento.
- E por fim, colocar a plataforma em execução com o objetivo de coletar dados sobre o seu funcionamento.

Finalmente, gostaria de agradecer meu orientador Marcio Maia pela ajuda e compreensão ao longo desse 1 ano de projeto, e estender esse agradecimento ao meu amigo Felipe Pinho que tanto me ajudou nas horas difíceis.

Todos os códigos utilizados nesse projeto podem ser encontrados em: *https://github.com/cainammello*.

REFERÊNCIAS

AGARWAL, Y.; WENG, T. From buildings to smart buildings—sensing and actuation to improve energy efficiency. **IEEE Design & Test of Computers**, IEEE, vol.29, n. 4, p. 36–44, 2012.

ASHTON, K. That 'internet of things' thing. **RFiD Journal**, v. 22, n. 7, p. 97–114, 2009.

BLUETOOTH, S. The bluetooth core specification, v4. 0. **Bluetooth SIG: San Jose, CA, USA**, 2010.

BLUETOOTH, S. **Specification of the Bluetooth System-Covered Core Package version: 4.0**. Juni, 2010. Disponível em: <https://www.bluetooth.org/docman/handlers/downloaddoc.ashx?doc_id=229737>. Acesso em: 27 maio 2016.

COLLINA, M.; CORAZZA, G. E.; VANELLI-CORALLI, A. Introducing the qest broker: Scaling the iot by bridging mqtt and rest. In: IEEE. **2012 IEEE 23rd International Symposium on Personal, Indoor and Mobile Radio Communications-(PIMRC)**. [S.l.], 2012. p. 36–41.

CONTE, G.; MARCHI, M. D.; NACCI, A. A.; RANA, V.; SCIUTO, D. Bluesentinel: a first approach using ibeacon for an energy efficient occupancy detection system. In: **BuildSys@ SenSys**. 1st ACM International Conference on Embedded Systems For Energy-Efficient Buildings (BuildSys) 2014: ResearchGate, 2014. p. 11–19.

COUNCIL, N. Six technologies with potential impacts on us interests out to 2025. **Disruptive Civil Technologies 2008**, 2008.

EVANGELATOS, O.; SAMARASINGHE, K.; ROLIM, J. Evaluating design approaches for smart building systems. In: IEEE. **Mobile Adhoc and Sensor Systems (MASS), 2012 IEEE 9th International Conference on**. Las Vegas, NV: IEEE, 2012. p. 1–7.

GARTNER. **Gartner's Hype Cycle**. Gartner, Inc., 2012. Gartner. Disponível em: <http://www.gartner.com/technology/research/hype-cycles/.>

GIGAOM. **With iBeacon, Apple is going to dump on NFC and embrace the internet of things**. -, 2016. Gigaom. Disponível em: <https://gigaom.com/2013/09/10/with-ibeacon-apple-is-going-to-dump-on-nfc-and-embrace-the-internet-of-things/>. Acesso em: 27 maio 2016.

GLUHAK, A.; KRCO, S.; NATI, M.; PFISTERER, D.; MITTON, N.; RAZAFINDRALAMBO, T. A survey on facilities for experimental internet of things research. **IEEE Communications Magazine**, v. 49, n. 11, p. 58–67, 2011.

GOMEZ, C.; OLLER, J.; PARADELLS, J. Overview and evaluation of bluetooth low energy: An emerging low-power wireless technology. **Sensors**, Molecular Diversity Preservation International, v. 12, n. 9, p. 11734–11753, 2012.

GUBBI, J.; BUYYA, R.; MARUSIC, S.; PALANISWAMI, M. Internet of things (iot): A vision, architectural elements, and future directions. **Future Generation Computer Systems**, Elsevier, v. 29, n. 7, p. 1645–1660, 2013.

HUI, J. W.; CULLER, D. E. Extending ip to low-power, wireless personal area networks. **Internet Computing, IEEE**, IEEE, v. 12, n. 4, p. 37–45, 2008.

HUNKELER, U.; TRUONG, H. L.; STANFORD-CLARK, A. Mqtt-s—a publish/subscribe protocol for wireless sensor networks. In: IEEE. **Communication systems software and middleware and workshops, 2008. comsware 2008. 3rd international conference on**. [S.l.], 2008. p. 791–798.

KARYDIS, T. **Bluetooth Interfacing with HM-10**. 2015. Disponível em: <http://fab.cba.mit.edu/classes/863.15/doc/tutorials/programming/bluetooth.html>. Acesso em: 17 outubro 2016.

KHOO, K. C. Y. **Automation of internet of things connection**. Tese (Doutorado) — UTAR, 2015.

KORTUEM, G.; KAWSAR, F.; FITTON, D.; SUNDRAMOORTHY, V. Smart objects as building blocks for the internet of things. **Internet Computing, IEEE**, IEEE, v. 14, n. 1, p. 44–51, 2010.

LAMPKIN, V.; LEONG, W. T.; OLIVERA, L.; RAWAT, S.; SUBRAHMANYAM, N.; XIANG, R.; KALLAS, G.; KRISHNA, N.; FASSMANN, S.; KEEN, M. et al. **Building smarter planet solutions with mqtt and ibm websphere mq telemetry**. [S.l.]: IBM Redbooks, 2012.

LANDT, J. The history of rfid. **IEEE potentials**, IEEE, Los Alamos Nat. Lab., NM, USA, v. 24, n. 4, p. 8–11, 2005.

LEE, B.-u.; IM, S.-y.; LEE, S.-w.; KIM, B.; ROH, B.-h.; KO, Y.-B. The beacon identification using low pass filter for physical web based iot services. In: IEEE. **Communications, Computers and Signal Processing (PACRIM), 2015 IEEE Pacific Rim Conference on**. Victoria, BC: IEEE, 2015. p. 354–358.

MQTT, O. **MQTT: MQ Telemetry Transport**. 2016.

ORGANIZATION, M. **MongoDB Architecture**. 2016.

PEREIRA, C. R. **Aplicações web real-time com Node. js**. [S.l.]: Editora Casa do Código, 2014.

PETTEY, C.; MEULEN, R. Van der. **Gartner's 2012 hype cycle for emerging technologies identifies"Tipping Point"technologies that will unlock long-awaited technology scenarios**. Gartner to Host Complimentary Webinar: Newsroom, 2012.

RIVERA, J.; MEULEN, R. v. d. **Hype Cycle for Emerging Technologies, 2015**. Gartner, Inc., 2015. Gartner. Disponível em: <http://www.gartner.com/newsroom/id/3114217>. Acesso em: 27 setembro 2016.

SNOONIAN, D. Smart buildings. **Spectrum, IEEE**, IEEE, v. 40, n. 8, p. 18–23, 2003.

STATLER, S. Geofencing: Everything you need to know. In: **Beacon Technologies**. [S.l.]: Springer, 2016. p. 307–316.

TAKALO-MATTILA, J.; KILJANDER, J.; SOININEN, J.-P. Advertising semantically described physical items with bluetooth low energy beacons. In: IEEE. **Embedded Computing (MECO), 2013 2nd Mediterranean Conference on**. 2013 2nd Mediterranean Conference on Embedded Computing (MECO), Budva: IEEE, 2013. p. 211–214.

WANT, R. An introduction to rfid technology. **IEEE Pervasive Computing**, IEEE, IEEE Pervasive Computing, v. 5, n. 1, p. 25–33, 2006.

XAPPSOFTWARE. **is MQTT the panacea for all the problems of IoT?** -, 2014. XAppSoftware Blog. Disponível em: <http://www.xappsoftware.com/wordpress/2014/10/20/is-mqtt-the-panacea-for-all-the-problems-of-iot/>. Acesso em: 17 outubro 2016.

ZAFARI, F.; PAPAPANAGIOTOU, I. Enhancing ibeacon based micro-location with particle filtering. Globecom, 2015.

ZAFARI, F.; PAPAPANAGIOTOU, I.; CHRISTIDIS, K. Micro-location for internet of things equipped smart buildings. IEEE, 2015.