UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
ESCOLA DE ENGENHARIA
TRABALHO DE CONCLUSÃO EM ENGENHARIA DE CONTROLE E AUTOMAÇÃO

Sistema de Localização Indoor em Tempo Real

Autor: Maurício Dall'Oglio Farina

Orientador: Prof. Edison Pignaton de Freitas

Porto Alegre, 5 de janeiro de 2018

Sumário

Resumo **ii**

Abstract **iii**

Lista de Figuras **iv**

Lista de Tabelas **v**

Lista de Abreviaturas e Siglas **vi**

Lista de Símbolos **vii**

1 Introdução **1**
1.1 Objetivos 4

2 Revisão bibliográfica **6**
2.1 RTLS 6
2.2 Bluetooth Low Energy 8
2.3 RSSI 9
2.4 MQTT 9

3 Metodologia **10**
3.1 Framework 10
3.1.1 Padronização 12
3.1.2 Transmissão 14
3.1.3 Análise 16
3.1.4 Integração 16
3.2 Sistema de Localização Indoor 18
3.2.1 Requisitos do Sistema 18
3.2.2 Materiais 18
3.2.3 Softwares 19
3.2.4 Considerações iniciais 19
3.2.5 Instalações e Configurações 19
3.2.6 Configuração do *framework* 20
3.2.7 Algoritmo 20

4 Resultados **22**
4.1 Algoritmo 22
4.2 Filtros 25

4.3 Média Quadrática 26
4.4 Verificação da qualidade das estimativas 26

5 Conclusões **28**

Referências **29**

Resumo

Devido à disrupção das tecnologias de comunicação móvel e à proliferação das redes sem fio, tornou-se possível a implementação de sistemas de rastreamento. Neste contexto os sistemas de localização *Real-Time Location System* (RTLS) despontam como importantes agentes no monitoramento de processos, controle de ativos, controle de fluxo de pessoas e sistemas de auto-localização. No entanto, a implementação de sistemas de localização pode ainda representar um alto investimento monetário e se depara com dificuldades de integração multi-plataforma e complexabilidade de utilização. Neste trabalho foi desenvolvida um *framework* para um sistema de localização com a finalidade de estabelecer-se uma plataforma universal para todo possível escopo de diferentes leitores, padrões de mensagem, aplicações e estruturas de armazenamento. Além disso, implementou-se um sistema de localização *indoor* para validação deste *framework*. Por fim, foram realizados testes, cujos resultados foram analisados a fim de se verificar a qualidade das estimativas obtidas. Finalmente, concluiu-se que os valores obtidos eram satisfatórios para o bom funcionamento do sistema.

Abstract

Due to the spreading of wireless networks and the desruption of mobile communication technologies, the implementation fo tracking systems was made possible. For that reason, Real-Time Location Systems (RTLS) stand out as an important solution for process management, asset control, people flow control and auto localization systems. However, the implementation of a RTLS may still require a large finacial investment, face cross-plataform integration dificulties and a greate complexibility. In this paper, a RTLS framework was developed in order to obtain a cross-plataform solution between devices, messages protocols, applications and data structures. Besides that, a indoor RTLS was developed in order to volidate this framework. Finally, the solution was tested and it's results analized, so that the quality of the results could be verified. At last, it was concluded that the obtained results were good enough for the well behavior of the system.

Lista de Figuras

1 Meio Portal de estrutura *choke-points* 2
2 Leituras cruas do sinal de *Received Signal Strength Indication* (RSSI) de um portal para os 3 canais do *Bluetooth Low Energy* (BLE) 2
3 Resultado do processamento, definição da localização e geração de eventos para um portal 3
4 Módulos do *framework* 4
5 Estrutura clássica de um sistema de localização em tempo real 10
6 Módulo Padronização 13
7 Módulo Transmissão 15
8 Módulo Análise 16
9 Módulo Integração 17
10 Software SGF 17
11 Leitor Bluecats modelo Edge (Esquerda) e Tags (Direita) 18
12 Planta baixa do ambiente e disposição dos equipamentos 20
13 Fluxograma do algoritmo 21
14 Planta baixa com a disposição dos equipamentos e *tag* do exemplo apresentado . . 22
15 Sinal de RSSI lido pelo leitor e sinal calculado a partir da média dos últimos dois valores 25
16 Qualidade de acertos para *tags* com potência de sinal de -20 dB 26
17 Qualidade de acertos para *tags* com potência de sinal de -30 dB 27

Lista de Tabelas

1 Graus de precisão 8
2 Lista de equipamentos, *softwares* e soluções utilizadas no desenvolvimento da *framework* 11
3 *Mensagem Generic Input Message Object* (GIMO) 12
4 Convenção de nomenclatura para arquivos de classes de leitores 13
5 Mensagem *Generic Output Message Object* (GOMO) 16
6 Perda de sinal para diferentes materiais 19
7 Leituras com tempos inferiores a 5 segundos 23
8 Contagem de leituras por leitor 24
9 Seleção das últimas duas leituras mais recentes do leitor 1 24
10 Seleção das últimas duas leituras mais recentes do leitor 2 24
11 Seleção das últimas duas leituras mais recentes do leitor 3 24
12 Seleção das últimas duas leituras mais recentes do leitor 4 24
13 Ranking final dos leitores 25

Lista de Abreviaturas e Siglas

BLE	*Bluetooth Low Energy*. iv, 2
CLT	Consolidação das Leis do Trabalho. 1
FIR	*Finite Impulse Response*. 25
GDO	*Generic Device Object*. 13, 14
GIMO	*Generic Input Message Object*. 12–14
GO	*Generic Objects*. 13
GOMO	*Generic Output Message Object*. 16
IDE	*Integrated Development Environment*. 19, 20
IoT	*Internet of Things*. 9
MQTT	*Message Queue Telemetry Transport*. 9, 10, 20
RDO	*Real Device Object*. 13, 14
RMO	*Real Message Object*. 13, 14
RO	*Real Objects*. 13
RSSI	*Received Signal Strength Indication*. iv, 2, 9, 21, 25, 26
RTLS	*Real-Time Location System*. ii, iii, 1, 4–8, 10, 16, 18

Lista de Símbolos

A intensidade recebida a 1 metro de distância em decibéis. 9

d distância em metros entre o leitor e o transmissor. 9

n expoente de perda de sinal. 9

1 Introdução

A disrupção da computação móvel e das redes sem fio fomentou o interesse do mercado em sistemas de *tracking* como um todo e dentre estes, sistemas de localização indoor em tempo real baseados em *beacons bluetooth*.

Segundo Malik 2009, nos tempos atuais é muito comum um indivíduo ter consigo pelo menos um dispositivo com múltiplas possibilidades de rastreamento. Neste contexto, a implementação de um sistema de *tracking* é extremamente aplicável e desponta como importante agente no monitoramento de processos, mercadorias e pessoas para as mais diversas necessidades. Como necessidades do mercado atual pode-se enunciar o controle de ativos, o controle de fluxo de pessoas, sistemas de auto-localização e também aplicações em sistemas de segurança.

Tendo em vista o benefício do desenvolvimento de uma solução simples, de baixo custo e flexível na utilização de RTLS buscou-se, neste trabalho, a composição de uma plataforma universal para implementação destes, utilizando-se um formato padronizado e de fácil integração.

Soluções de localização em tempo real baseados em *beacons bluetooth* podem ser encontradas hoje nos mais sofisticados formatos, apresentando excelente qualidade e precisão. No entanto, de acordo com a experiência do autor, estes sistemas tendem a ser caros, não inter-compatíveis e de baixa integração com sistemas e softwares de terceiros. Dentre estes diversos sistemas, pode-se citar como exemplo a solução implementada em uma das maiores companhias da industria de carnes do mundo. De acordo com a Consolidação das Leis do Trabalho (CLT) *Decreto-lei nº 5.452, Artigo nº 253* 1943 - Para os empregados que trabalham no interior das câmaras frigoríficas e para os que movimentam mercadorias do ambiente quente ou normal para o frio e vice-versa, depois de 1 (uma) hora e 40 (quarenta) minutos de trabalho contínuo, será assegurado um período de 20 (vinte) minutos de repouso, computado esse intervalo como de trabalho efetivo. Sendo assim, o sistema busca monitorar o tempo de permanência dos colaboradores nas áreas frias e quentes a fim de assegurar a correta execução das pausas térmicas. Essa solução baseia-se em uma estrutura de *choke-points* onde são instalados dois leitores, um em cada lado do portal (lado quente e lado frio). A partir da Figura 1, pode-se observar o equipamento instalado acima da porta. Este é composto de uma antena direcional (parte clara do equipamento) e um equipamento que realiza leituras (parte cinza do equipamento). Este equipamento mostrado, representa metade de um portal, o qual torna-se completo ao se levar em conta um segundo equipamento, idêntico ao apresentado na imagem, do outro lado da porta.

Figura 1: Meio Portal de estrutura *choke-points*

Baseando-se nas leituras captadas por antenas direcionais, o algoritmo estima um evento de entrada ou de saída pelo portal através dos pontos de cruzamento entre o RSSI dos leitores (Figura 2), gerando então eventos de passagem (Figura 3).

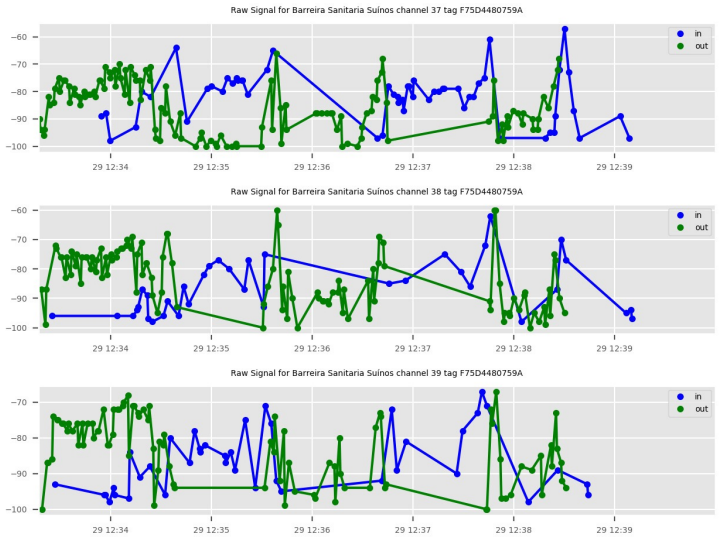

Figura 2: Leituras cruas do sinal de RSSI de um portal para os 3 canais do BLE

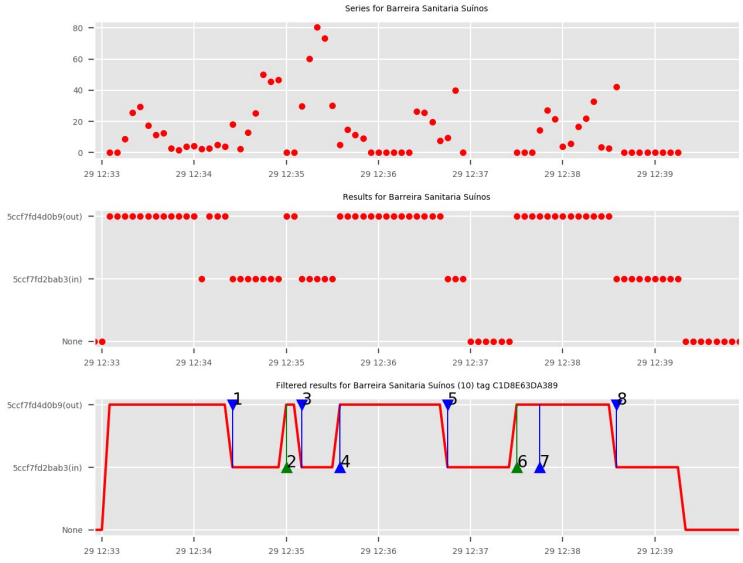

Figura 3: Resultado do processamento, definição da localização e geração de eventos para um portal

No entanto, devido à grande complexabilidade do algoritmo, é necessário um grande poder computacional para realizar a geração dos eventos, já que, o mesmo, faz uso de filtros com um grande número de coeficientes, além de, também, realizar cálculos considerando toda a trajetória passada da *tag*. Unindo-se isto à grande quantidade de *tags* em circulação (entorno de 15000), tem-se, como consequência, uma grande demanda computacional por parte do algoritmo. Além disso, este sistema apresenta uma grande dificuldade de instalação pois requere o alinhamento das antenas seguido de testes e ajustes locais do algoritmo, o que pode levar dias ou até semanas. Essa dificuldade somada ao longo tempo de desenvolvimento, acarreta em um custo final por portal instalado muito alto (na ordem de milhares de dólares por unidade), o que em muitos casos inviabiliza a implementação do sistema.

Uma outra solução, ainda não implementada, mas que já está sendo desenvolvida para um grande complexo hospitalar na cidade de Porto Alegre, é o gerenciamento de enfermeiras. Segundo Somensi 2016, os modelos de gerenciamento atuantes hoje no Brasil baseiam-se em modelos internacionais. Estes, desconsideram fatores importantes da realidade dos sistema de enfermagem brasileiro, uma vez que não diferenciam profissionais de nível médio e superior. Além disso, não levam em conta a área de atuação do enfermeiro, que comumente está habituado a realizar uma lista específica de procedimentos no seu cotidiano, tornando-se mais eficiente nestes em comparação com outro colega atuante em um setor diferente. Tendo em vista este problema, o grupo de pesquisa da entidade hospitalar decidiu realizar um estudo para resolvê-lo. Graças a isso, surgiu-se a necessidade de coleta de dados do dia-a-dia de cada enfermeiro que juntando-se a outras necessidades possibilitou o desenvolvimento de um sistema capaz de localizar enfermeiros em tempo real, analisar a demanda de serviço por setor e levantar dados estatísticos para o estudo em questão. Assim, o sistema possibilitará que a equipe de gestão realoque funcionários de um setor

para outro de acordo com os picos de demanda com mais facilidade e agilidade. Nesse contexto, também é válido mencionar a relevância do baixo custo dos equipamentos a serem instalados devido ao grande volume necessário.

Visto isso, torna-se clara a necessidade de uma solução simples, barata e flexível, a fim de abraçar uma fatia no mercado de RTLS, o que muitas vezes é inviável devido aos problemas mencionados anteriormente. Assim, buscou-se, neste trabalho, compor uma *framework* para sistemas de RTLS em um formato padrão, multi-plataforma e de fácil integração.

A estruturação dessa plataforma se deu por meio da criação de um *framework* que abstraiu as aplicações e os leitores reais transformando-os em aplicações e leitores genéricos e virtuais. Dessa forma, as particularidades de cada leitor, que antes representavam uma dificuldade de desenvolvimento, principalmente em sistemas multi-plataforma, deixaram de ser relevantes.

Por fim, utilizou-se a plataforma a fim de desenvolver-se um sistema de localização capaz de determinar o cômodo ao qual ativos e pessoas, portadores de uma *tag*, estão presentes. Este sistema, poderá então ser utilizado com a finalidade de encontrar-se objetos ou pessoas (como por exemplo médicos e equipamentos em um hospital), realizar-se um controle de acesso a locais restritos, realizar-se o controle de evasão de pessoas e objetos (como por exemplo um idoso em uma casa de repouso ou um produto em uma loja), levantar-se dados estatísticos a respeito do uso de equipamentos ou da movimentação de pessoal, dentre diversas outras.

1.1 Objetivos

O objetivo deste trabalho é desenvolver um *framework* para sistemas RTLS composto por quatro módulos: padronização, transmissão, análise e integração. Pela Figura 4, pode-se observar o fluxo ao qual as informações deverão seguir, também como a estrutura padrão do *framework* proposto. De maneira geral, o sistema inicia a partir do módulo Padronização, captando-se leituras. Em seguida, mensagens de entrada são enviadas do módulo Padronização a um módulo Transmissão (Transmissão (a)) o qual as armazena e as fornece ao módulo Análise. Após processar as informações fornecidas pelo módulo anterior, o módulo Análise envia mensagens de saída a um outro módulo Transmissão (Transmissão (b)), o qual as armazena e as fornece ao módulo Integração. Por fim, o módulo Integração traduz as mensagens de saída a fim de realizar a integração a um ou mais softwares de terceiros.

Figura 4: Módulos do *framework*

Em seguida, com o objetivo de demonstrar o seu funcionamento, deverá se utilizar o *framework* desenvolvido para se compor um pequeno sistema de localização *indoor*.

Padronização: No módulo Padronização, diferentes equipamentos de diferentes vendedores serão integralizados de forma a apresentar um comportamento padronizado. Para isto, será necessário que, para cada novo leitor, seja adicionado suporte no *framework*. Para tanto, deverá ser realizado o desenvolvimento de classes e métodos os quais serão responsáveis por configurar, inicializar, gerenciar e padronizar o comportamento e funcionamento do mesmo, a fim de que todos os leitores apresentem um comportamento padrão convencionado. Na sequência deste trabalho, serão apresentados as estruturas escolhidas e os padrões convencionados, os quais foram utilizados no desenvolvimento deste *framework*.

Transmissão: No módulo Transmissão, as mensagens de entrada ou saída recebidas serão unificadas e transmitidas por um canal único e selecionável. Para tanto, será necessário que se adicione suporte ao *framework* para cada novo banco de dados, protocolo de comunicação ou qualquer outro sistema de armazenamento e/ou comunicação a ser utilizado. No entanto, devido ao vasto universo de soluções existentes e possíveis de serem utilizadas, não será possível propor, neste trabalho, um padrão de desenvolvimento para o suporte de novas aplicações. Entretanto, no decorrer do texto, paradigmas de seu comportamento serão convencionados e apresentados, os quais deverão obrigatoriamente ser seguidos no desenvolvimento de novos suportes a plataforma.

Análise: No módulo Análise, será realizado o processamento das informações captadas (mensagens de entrada) e a geração de eventos de análise (mensagens de saída). Neste módulo, o desenvolvedor que utilizar o *framework* terá um espaço livre para o desenvolvimento de seu algoritmo. No entanto, a plataforma restringirá que toda mensagem de entrada e saída, do módulo Análise, deva ser obtida ou transmitida por a partir de um módulos Transmissão.

Integração: No módulo Integração, será realizado o desenvolvimento de suporte a aplicações de terceiros como *softwares* de gestão, aplicativos de interface ao usuário, sistemas de monitoramento e controle de pessoas e ativos e qualquer outra aplicação a qual for desejável ou útil as informações obtidas pelo sistema RTLS desenvolvido.

Essa modularização tem como objetivo facilitar o processo de desenvolvimento de novas soluções, também como a adição de novos equipamentos, sistemas de armazenamento e outros *softwares*. Além disso, a plataforma possibilita que tanto sistemas simples como complexos sejam desenvolvidos no mesmo ambiente. Como consequência, novos projetos serão concebidos de forma mais ágil e com tempo de desenvolvimento reduzido, o que gera uma redução de custos de projeto.

O trabalho apresentado a seguir inicia-se com um capítulo de Revisão Bibliográfica onde a teoria a respeito das tecnologias e metodologias englobadas neste trabalho são apresentadas. Na continuação é apresentada a Metodologia. Nesta inicialmente são abordados os diferentes módulos do *framework* e, em seguida, a implementação do sistema de localização. Por fim são apresentados os capítulos referentes aos Resultados e Conclusão respectivamente.

2 Revisão bibliográfica

A seguir, serão apresentados os principais assuntos e termos utilizados neste trabalho.

2.1 RTLS

Sistemas de localização em tempo real (*Real-Time Locating Systems* no inglês), mais conhecidos como RTLS, são sistemas locais de identificação e rastreamento da localização de ativos, e/ou pessoas em tempo real ou quase real (Boulos e Berry 2012). Malik 2009 também define um RTLS como um sistema que permite a localização, rastreamento, gerenciamento, análise, aproveite, e, de outra forma, utilize a informação a respeito da localização de ativos ou pessoas. Sistemas RTLS podem ser desenvolvidos em diversos graus de precisão e utilizar uma infinidade de tecnologias sem fio disponíveis no merca. Na comunicação, diversos protocolos de comunicação são comumente utilizados, dentre eles:

***Ultra Wide Band* (UWB):** esta é uma tecnologia de transmissão de dados sem fio que pode eventualmente vir a tornar-se o padrão dominante da indústria. Ao invés de operarem numa frequência fixa, os transmissores UWB utilizam um número quase infinito de frequências entre 0 e 60 GHz, sem permanecer em uma única frequência por mais do que algumas frações de segundo. Apenas as duas partes envolvidas conhecem o padrão de frequências utilizado, o que ajuda a manter a segurança dos dados (Morimoto 2017).

***Geo Positioning System* (GPS):** é um sistema de radionavegação por satélite desenvolvido e operado pelo Ministério da Defesa dos E.U.A. O GPS permite determinar a posição, velocidade e o fuso horário dos utilizadores em terra, mar e aerotransportados 24 horas por dia, em todas as condições climáticas e em qualquer parte do mundo. Os sinais GPS são disponibilizados simultaneamente para um número ilimitado de utilizadores. Cada satélite do sistema GPS transmite sinais para equipamentos no solo. Os receptores GPS recebem passivamente os sinais provenientes dos satélites, mas não transmitem. O sistema GPS usa uma rede de satélites que permite assinalar a localização de pessoas com receptores GPS em qualquer parte do mundo (TomTom 2017).

LoRa: a rede LoRa é uma solução sem fio sub-GHz em frequência não licenciada que endereça demandas para conexão entre dispositivos para aplicações de baixo consumo, longa distância (em alguns locais é possível conseguir 15 km) e baixo custo de infra-estrutura, considerando-se o grande número de nós. Como é possível verificar abaixo, a rede LoRa é bem popular e tem tido bastante aderência ao redor do mundo (Nunes 2017).

***Radio-Frequency IDentification* (RFID):** essa tecnologia nada mais é do que um termo genérico para as tecnologias que utilizam a frequência de rádio para captura de dados. Por isso existem diversos métodos de identificação, mas o mais comum é armazenar um número de série que identifique uma pessoa ou um objeto, ou outra informação, em um microchip. Tal tecnologia

permite a captura automática de dados, para identificação de objetos com dispositivos eletrônicos, conhecidos como etiquetas eletrônicas, *tags*, *RF tags* ou *transponders*, que emitem sinais de radiofrequência para leitores que captam estas informações. Ela existe desde a década de 40 e veio para complementar a tecnologia de código de barras, bastante difundida no mundo. A sua principal função hoje não é simplesmente substituir o código de barras, pois ela é uma tecnologia de transformação que pode ajudar a reduzir desperdício, limitar roubos, gerir inventários, simplificar a logística e aumentar a produtividade. Uma das maiores vantagens dos sistemas baseados em RFID é o fato de permitir a codificação em ambientes hostis e em produtos onde o uso de código de barras não é eficaz (Castro 2017).

***Zigbee*:** segundo Betters e Hall 2017, o protocolo *ZigBee* baseia-se no padrão IEEE 802.15.4 para redes pessoais. Este, encontra-se presente em sistemas de automação a mais de uma década, e é amplamente considerado como uma alternativa ao *Wi-Fi* e *Bluetooth* em situações em que é necessário um baixo consumo de energia e um pequeno tráfego de dados. Este protocolo é fortemente utilizado devido ao fato de que, dispositivos *ZigBee-enabled* funcionam e operam em conjunto, ao mesmo tempo que possibilitam o controle dos mesmos. Um típico exemplo de aplicação do protocolo são produtos como interruptores e lâmpadas *smart*, os quais, mesmo sendo de fabricantes diferentes, comunicam-se a fim de se controlar o acendimento da lâmpada.

***Low Power Bluetooth (Bluetooth LE, BLE or Bluetooth 4.0)*:** este protocolo será apresentado em mais detalhes no seguimento desta seção.

WLAN (WIFI): o IEEE 802.11 é um padrão internacional que descreve as características de uma rede local sem fio (WLAN). O nome Wi-Fi (contração de *Wireless Fidelity*) corresponde, inicialmente, ao nome dado à certificação emitida pela *Wi-Fi Alliance*, antigamente WECA (*Wireless Ethernet Compatibility Alliance*), o organismo encarregado de manter a interoperabilidade entre os hardwares que respondem à norma 802.11. Por razões de marketing, ou abusos de linguagem, o nome da norma é confundido, hoje, com o nome da certificação. Assim, uma rede Wi-Fi é realmente uma rede que responde à norma 802.11 (CCM 2017).

Cada uma dessas tecnologias apresenta suas particularidades. Sendo assim, a escolha da mesma deve ser feita criteriosamente de acordo com o nível de precisão desejado para o RTLS. Dentre os níveis de precisão, pode-se observar uma breve descrição na Tabela 1.

Tabela 1: Graus de precisão

Nível	Descrição
Presença	O sistema retorna que o ID está presente ou não (Cobertura para grandes áreas).
Sala	O sistema retorna em qual sala o ID está presente.
Sub-Sala	O sistema informa em qual lugar especifico o ID se encontra dentro de uma sala, pode-se usar como exemplo, um quarto de hospital com vários leitos.
Portal	O sistema detecta quando um evento de passagem acontece em um portal (porta, cancela, corredor, dentre outros), retornando o ID do evento juntamente com a direção da transição.
Associação	Um ID retorna a sua localização baseado na localização de outro ID.
Preciso	O sistema estima as coordenadas globais da localização de um ID e retorna as suas coordenadas.

Resumidamente, um sistema de RTLS baseia-se em leitores ou receptores dispostos em um ambiente externo, interno ou híbrido conhecido que recebem sinais provindos de pequenos dispositivos chamados de *tags*. Com isso, determina-se a posição aproximada utilizando-se algum algoritmo de calculo de posicionamento. Como principal exemplo de RTLS tem-se o sistema global de posicionamento (GPS) o qual é hoje difundido internacionalmente e que tornou-se vital no cotidiano do homem moderno.

Vale lembrar que um sistema RTLS não necessariamente inclui detalhes completos ou contínuos de navegação como velocidade, direção ou orientação espacial do ativo ou pessoa a ser rastreado (Boulos e Berry 2012).

2.2 Bluetooth Low Energy

O protocolo de comunicação *Bluetooth Low Energy*, também conhecido como *Bluetooth LE, BLE ou Bluetooth 4.0*, foi concebido com o objetivo de interconectar dispositivos à curta distância e com um baixo custo energético, requisito essencial hoje devido a disruptura da internet das coisas (IoT). Segundo Casaroli 2017, módulos BLE consomem em média de 5 a 15mA quando recebendo e 4uA quando em stand-by conectado. Em contraste com os módulos Wi-Fi, que em geral consomem pelo menos 50mA quando recebendo e pelo menos 400uA em stand-by conectado. Como resultado, um *beacon BLE* alimentado por 2 pilhas AA pode permanecer ativo por vários anos sem a necessidade de troca de baterias. Existem três tipos de modos de comunicação *Bluetooth*, chamados de *transiver class* 1, *class* 2 e *class* 3, no qual o alcance de comunicação é 100, 10, 5 m, respectivamente (Li et al. 2013).

Segundo Heydon 2013, ao contrário do *Bluetooth* clássico, o qual possui 40 canais, o BLE possui

apenas 3 canais chamados de 37, 38 e 39 (2402 MHz, 2426 MHz e 2480 MHz respectivamente). Isso com o objetivo de reduzir o consumo de energia. Tigoe 2017 explica que as duas principais alterações do *bluetooth* clássico para o 4.0 são a mudança para uma uma comunicação assíncrona, de forma que o rádio está ligado apenas quando o o periférico está transmitindo informações. Além disso, os dados não são apagados quando são lidos, mas sim quando são atualizados, fazendo com que múltiplos receptores possam acessar a mesma informação. Outra vantagem do BLE é o preço, um pequeno transmissor pode ser encontrado com preços abaixo de 2 dólares.

2.3 RSSI

O Indicador de intensidade de sinal recebido RSSI, como já diz o nome, é um indicador da intensidade recebida pelo rádio. A norma "IEEE 802.11" 2016 define que o RSSI é a relação entre intensidade transmitida e recebida na unidade dBm (Rida et al. 2015). Em uma área aberta e sem obstáculos, a potência do sinal recebido decai exponencialmente com a distância. Graças a essa propriedade, é possível estimar-se a distância entre o transmissor e o receptor pela equação 1.

$$RSSI = -10 \cdot n \cdot \log_{10} d + A \tag{1}$$

Aonde n é o expoente de perda de sinal, d é a distância entre o leitor e o transmissor, em metros, e A é a intensidade de sinal, em dB, recebida a 1 metro de distância.

Assim, para se estimar a distância, deve-se realizar um procedimento de calibração aonde coloca-se o transmissor e o receptor a distância de 1 metro, em seguida, mede-se o valor do RSSI, o qual resulta no valor de A. Em seguida, afasta-se a distância para 2 metros e mede-se o novo valor de RSSI. Substituindo-se então, na equação 1, d pelo valor da distância de 2 metros, A pelo valor de RSSI obtido na medição anterior, e RSSI pelo valor de RSSI medido nesta última medição, pode-se obter o valor do expoente de perda de sinal n. Por fim, substituindo-se A e n pelos valores encontrados na calibração, pode-se calcular a distância para qualquer outro valor de RSSI medido.

2.4 MQTT

O protocolo de comunicação *Message Queue Telemetry Transport* (MQTT) é um dos protocolos mais utilizados para a *Internet of Things* (IoT). O protocolo de mensagens MQTT é projetado para um baixo consumo de banda de rede e requisitos de hardware sendo extremamente simples e leve. O MQTT foi desenvolvido pela IBM e *Eurotech* e é projetado para enviar dados através de redes intermitentes ou com baixa banda de dados, para isto o protocolo é desenvolvido em cima de vários conceitos que garantem uma alta taxa de entrega das mensagens (Vicenzi 2017). De forma simplificada, o protocolo funciona a partir de um paradigma de *publish/subscribe* o qual, o *publishers* são responsáveis por publicar mensagem em tópicos de sua escolha, já os *subscribers* são capazes de realizar a assinaturas de tópicos a fim de receber as mensagens publicadas nos mesmos.

3 Metodologia

O trabalho realizado divide-se em duas partes: desenvolvimento de um *framework* e a utilização do mesmo para a implementação de um sistema RTLS. Primeiramente será apresentada a metodologia de desenvolvimento do *framework* bem como sua convenção. Em um segundo momento, fez-se uso do mesmo a fim de exemplificar e validar o seu funcionamento.

3.1 Framework

Como mencionado anteriormente, o *framework* desenvolvido compõem-se de quatro módulos. Essa divisão busca separar os diversos elementos que compõem um sistema de RTLS, como leitores, estruturas de armazenamento de dados e algoritmos de processamento das informações. O motivo ao qual escolheu-se modularizar-se o sistema da maneira proposta pode ser entendida a partir de um exemplo de modelo clássico de um sistema RTLS. Na Figura 5, leitores, da marca Bluecats, comunicam-se ao algoritmo de estimativa de posição pelo protocolo MQTT. Após os dados serem processados pelo algoritmo, os resultados são salvos em uma estrutura de banco de dados SQL Server a qual é disponibilizada ao sistema SAP. Comparando-se, então, a estrutura apresentada na Figura 4 ao exemplo da estrutura clássica, fica evidente a semelhança entre as partes do sistema real e os módulos do *framework*.

Figura 5: Estrutura clássica de um sistema de localização em tempo real

Devido a grande variedade de equipamentos e *softwares* disponíveis no mercado, foi necessário limitar-se a alguns destes para o desenvolvimento do *framework*. Com o objetivo de que inicialmente, a plataforma, tivesse um maior alcance e utilização por parte de outros desenvolvedores, selecionou-se soluções e equipamentos fortemente consolidados no mercado e que estavam disponíveis ao autor deste trabalho. A Tabela 2 apresenta os equipamentos, sistemas de armazenamento e soluções utilizadas no desenvolvimento da plataforma. É interessante ressaltar que, estes, não foram apenas utilizados no desenvolvimento base da *framework*, mas também foram adicionados a mesmo, fazendo com que todos já estejam sendo suportados e possam ser utilizados no desenvolvimento de novas soluções de RTLS.

Tabela 2: Lista de equipamentos, *softwares* e soluções utilizadas no desenvolvimento da *framework*

Logo	Marca	Modelo	Módulos	Tipo
	Bluecats	Edge	Padronização	Leitor
	Dev Tecnologia	DevSmartScanner	Padronização	Leitor
	Estimote	ERock	Padronização	Leitor
	Kontakt.io	Gateway	Padronização	Leitor
	OpenSource (IBM)	MQTT	Padronização	Protocolo
	OpenSource (IBM)	MQTT	Transmissão	Protocolo
	OpenSource	Noble	Padronização	Protocolo
	Kontakt.io	Kontakt Cloud	Padronização	Banco de Dados
	MariaDB Foundation	MariaDB	Transmissão	Banco de Dados
	Oracle	MySQL	Transmissão	Banco de Dados
	Microsoft	SQL Server	Transmissão	Banco de Dados
	Hwaci	SQLite	Transmissão	Banco de Dados
	-	Text File	Transmissão	Arquivo de Texto
	Exceed Solutions	SGF	Integração	Software
	SAP	SAP IoT	Integração	Software

A seguir, será abordado a estrutura e o processo de desenvolvimento para cada módulo do *framework* em questão. Além disso, serão apresentados os padrões escolhidos e as convenções definidas nas situações as quais se fazem necessários. Também, faz-se necessário ressaltar que toda a *framework* foi desenvolvida na linguagem JAVA devido ao fato de ser portável ao diversos sistemas operacionais possíveis de serem utilizados como *Windows, Linux, Mac OS X*, dentre outros. Para o desenvolvimento, utilizou-se orientação por objetos, teoria base desta linguagem de programação (Oracle 2017).

3.1.1 Padronização

Diferentes leitores de diferentes marcas tendem a apresentar diferenças significativas nos seus modos de configuração, operação e utilização. Isso se deve à inexistência de um padrão que regulamente esses dispositivos. Sendo assim, cada fabricante estabelece seu próprio modelo, o que dificulta a implementação de um sistema multi-plataforma. O módulo Padronização tem como objetivo resolver, então, esse problema. A padronização das leituras de diferentes leitores inicia obtendo-se a leitura específica de cada tipo de leitor. Para tanto, subdividiu-se o funcionamento do leitor em três partes: inicialização do leitor, aquisição das leituras e envio de mensagens.

Em geral, antes que se possa realizar a utilização e a extração de informações de um leitores, é necessário que sejam feitas configurações básicas de inicialização para o funcionamento do mesmo. Como exemplo, pode-se citar configurações de rede local (física ou sem fio), acesso a *internet*, escolha do protocolo de comunicação ou banco de dados, conexão com sistema na nuvem, padrão de mensagem, pre-processamentos e adição de filtros, períodos de coletas de dados, intensidade de sinal, escolha de canais, entre outros. Devido a isso, na inicialização do módulo, realizam-se as configurações iniciais necessárias para que o leitor em questão torne-se ativo e operante. Após este passo concluído, o equipamento iniciará a captação dos sinais transmitidos pelos *beacons* e os tornará disponíveis pele meio selecionado, como, por exemplo, através de protocolos de comunicação, requisição por um método, acesso a banco de dados, dentre outros.

É necessário então que inicie-se a segunda parte do módulo, a qual será responsável por realizar aquisição das leituras. Nesta etapa, as leituras serão extraídas (de onde quer que estejam sendo disponibilizadas) e, em seguida, traduzidas ao formato padrão *Generic Input Message Object* (GIMO), convencionado pelo autor. A definição desse formato, baseou-se nos diversos tipos de mensagens de leitura de diversos fabricantes, e assim, definiu-se que cada mensagem genérica GIMO deve ser composta das informações apresentadas na Tabela 3:

Tabela 3: *Mensagem Generic Input Message Object* (GIMO)

Informação	Definição
scannerMac	MAC Address do Leitor
tagMac	MAC Address da Tag
tagMeasuredPower	RSSI a 1 metro de distância
rssi	RSSI
rssiSmouth	RSSI Filtrado
timestamp	Data da leitura
data	Informação transmitida pela tag
battery	Porcentagem de bateria da tag
upTime	Tempo de funcionamento contínuo do leitor

Por fim, realiza-se a terceira e última etapa do módulo. Para isso, o módulo Padronização

deve realizar a comunicação e o envio das mensagens a um módulo Transmissão. O processo de envio dessas informações será de extrema simplicidade para o módulo Padronização, já que a abstração do envio das mensagens será realizada pelo módulo Transmissão o qual será apresentado posteriormente.

A Figura 6 ilustra uma representação gráfica da estrutura do módulo Padronização.

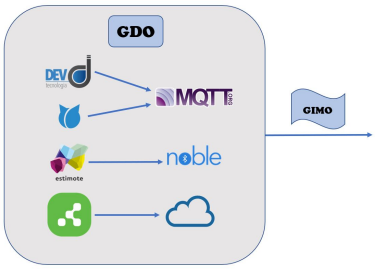

Figura 6: Módulo Padronização

Programação:

Para a implementação, do módulo Padronização, será necessário a criação de duas classes denominadas de *Real Message Object* (RMO) e *Real Device Object* (RDO). A classe RMO terá em seu conteúdo todas as informações contidas na mensagem com formato do fabricante. Já a classe RDO será responsável pela inicialização do leitor, aquisição de mensagens e, por fim, criação de objeto RMO. Este conjunto de classes será denominado de *Real Objects* (RO).

Com o objetivo de realizar-se a padronização dos diversos RO, o *framework* deve os abstrair utilizando duas classes denominadas GIMO e *Generic Device Object* (GDO) com a finalidade de se criar um leitor genérico virtual.

Este conjunto de classes será denominado de *Generic Objects* (GO). Ao ser instanciado, um objeto GIMO, recebe como atributo um objeto RMO e o converte para o formato padrão. Já o objeto GDO, é responsável abstrair e unificar a inicialização de leitores reais, criar objetos GIMO e conectar os módulos Padronização e Transmissão.

Também, definiu-se uma convenção de nomenclatura para as classes RDO e RMO exemplificada pela Tabela 4.

Tabela 4: Convenção de nomenclatura para arquivos de classes de leitores

Item	Nomenclatura
RDO	Marca_Modelo
RMO	Marca
Package	scanner.marca

Por fim, para realizar-se a adição de um novo leitor ao *framework*, deve-se realizar os seguintes passos:

1 - Criar as classes RMO e RDO

2 - Adicionar suporte ao GDO para o novo RDO

3 - Adicionar um novo construtor ao GIMO para que este suporte o novo RMO

3.1.2 Transmissão

O módulo Transmissão tem como objetivo abstrair todos os processos que se refiram a configuração, acesso, armazenamento e extração de dados dos diversos tipos de estruturas de armazenamento de dados as quais são suportadas pelo *framework*. Para tanto, o módulo pode ser subdividido em duas partes: inicialização do módulo e comunicação.

Ao se fazer uso de um novo módulo Transmissão, deve-se realizar-se a escolha da estrutura de armazenamento de dados a qual será utilizada pelo mesmo. A partir disto, o módulo realizará as configurações necessárias para que estrutura desejada fique pronta para receber as informações. Feita esta inicialização, o módulo deve abstrair o envio e recebimento de mensagens, de forma que seja unicamente possível realizar-se estas trocas de informação por intermédio do próprio módulo. Assim, o mesmo módulo de Transmissão poderá ser utilizado pelos módulos de Padronização, Análise e Integração a fim de enviar e receber mensagens.

Para a realização da abstração da estrutura de armazenamento, é de extrema complexabilidade definir uma estrutura padrão entre bancos de dados, protocolos, arquivos, dentre outros, já que apresentam grandes particularidades entre si. Dessa forma, a única definição proposta neste trabalho é que as mensagens devem ser salvas separadamente para cada *tag*, garantindo que o acesso às suas informações seja rápido. Alguns exemplos de estruturas de armazenamento de dados serão apresentados a seguir:

Bancos de dados: inicialmente foi inserido suporte para bancos de dados *MySQL*, *SQL Server*, *SQLLite* e *MariaDB* ao *framework*. Para cada nova *tag* identificada, uma nova tabela deve ser criada utilizando-se o seu *MAC Address* como nome. Além disso, quando deseja-se extrair mensagens, o processo de seleção das leituras relacionadas à cada *tag* considera apenas os últimos valores adicionados à tabela (valor ajustável e inicialmente definido como 1000). Essas definições tendem a melhorar a eficiência dos tempos de envio e extração de dados, o que é um fator crítico para sistemas de *RTLS*.

Arquivos Texto: para que o módulo Transmissão opere com arquivos de texto simples, definiu-se que para cada nova *tag*, uma nova pasta deve ser criada utilizando-se o *MAC Address* do leitor. Dentro desta são armazenadas as mensagens nos formatos *CSV* ou JSON em arquivos de texto simples com a extensão *.data*. Além disso, os arquivos organizam as mensagens por data de ocorrência, sendo nomeadas no padrão $[MACAddress]\_[Ano\_Mes\_Dia].data$. Por fim, para

realizar-se a extração de mensagens de um desses arquivos, uma cópia do arquivo em questão será feita e, esta, será utilizada para acessar as mensagens.

Protocolo *MQTT*: devido à necessidade de uma forma rápida e eficiente de envio e recebimento de mensagens, a qual os bancos de dados não são capazes de alcançar, utilizou-se o protocolo de comunicação *MQTT*. Basicamente, toda mensagem publicada no *MQTT Broker* é armazenada em um tópico nomeado $generic/[MAC\_Address\_da\_Tag]$. O retentor mantém no tópico apenas as últimas publicações (ajustável e inicialmente definido como 1000). Grande parte dos leitores comerciais utilizam o protocolo *MQTT* para o envio de suas mensagens, por ser um protocolo leve, rápido e estável. Por esta razão, o mesmo será definido como aplicação padrão de comunicação para o *framework* projetada neste trabalho.

Programação:

Para a implementação do módulo, desenvolveu-se uma classe denominada de *Bridge*, com a finalidade de realizar a abstração desejada. Para tando, a mesma é composta de um método construtor, um método de envio e um método de recebimento. Ao se instanciada a classe, o método construtor realizará o processo de inicialização da estrutura de armazenamento de dados escolhida. Como resultado, o objeto instanciado será o próprio módulo Transmissão. No método de recebimento, mensagens serão recebidas pelo método como atributo e armazenadas de acordo com a estrutura selecionada. Por fim, no método de envio, retornará as mensagens referentes a *tag* requisitada.

A Figura 7 apresenta uma representação gráfica da estrutura do módulo Transmissão.

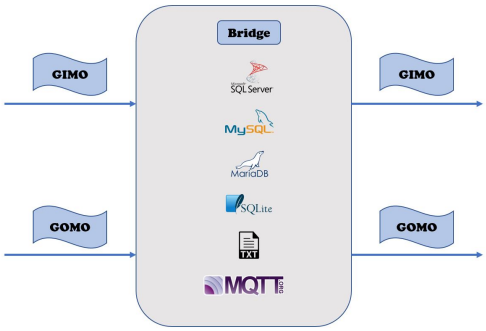

Figura 7: Módulo Transmissão

Por fim, para realizar-se a adição de uma nova estrutura de armazenamento de dados ao *framework* deve-se realizar os seguintes passos:

1 - Adicionar o processo de inicialização ao construtor na classe *Bridge* para a estrutura desejada

2 - Adicionar o processo de envio de mensagens ao método de envio na classe *Bridge* para a estrutura desejada

3 - Adicionar o processo de recebimento de mensagens ao método de recebimento na classe *Bridge* para a estrutura desejada

3.1.3 Análise

O módulo Análise tem como função processar as mensagens das leituras na informação desejada. Fica a critério do desenvolvedor projetar esse algoritmo da forma que o desejar. No entanto, toda informação de entrada e saída de leituras deve ser realizada pelo uso de módulos Transmissão.A Figura 8 apresenta uma representação gráfica da estrutura do módulo Análise.

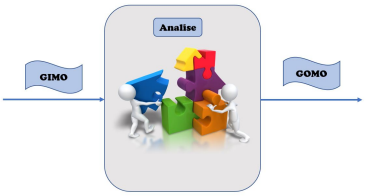

Figura 8: Módulo Análise

Além disso, convencionou-se um padrão para a mensagem de saída, implementado em uma classe denominada *Generic Output Message Object* (GOMO). A Tabela 5 apresenta o padrão convencionado para as mensagens de saída.

Tabela 5: Mensagem *Generic Output Message Object* (GOMO)

Informação	Definição
locationID	ID do local a qual a tag se encontra
coordinates	Coordenadas da tag
tagMac	MAC Address da Tag
timestamp	Data de criação do evento
data1	Campo livre
data2	Campo livre
data3	Campo livre
data4	Campo livre
data5	Campo livre
data6	Campo livre
battery	Porcentagem de bateria da tag

3.1.4 Integração

O Módulo Integração consiste em integrar os resultados obtidos no módulo Análise a softwares de terceiros. Estes *softwares* terão a finalidade de utilizar os resultados obtidos pelo sistema RTLS

na forma de interface para o usuário, realizar o controle de acesso de ativos ou pessoas, gerar levantamentos estatísticos, entre outros. Na integração entre sistemas, cada sistema, ou grupo de sistemas, é um caso único. Por isso, torna-se necessário o desenvolvimento de um novo algoritmo para cada novo *software* adicionado ao *framework*. Portanto, o módulo Integração será uma pseudo biblioteca de algoritmos de integração. Sua estrutura pode ser vista na Figura 9.

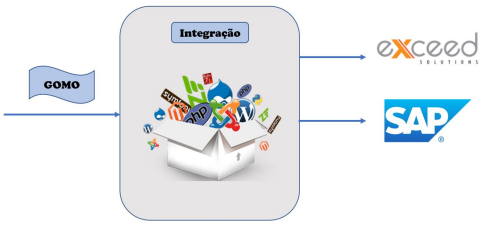

Figura 9: Módulo Integração

Pode-se citar como exemplo a adição de suporte ao *software SGF* da empresa *Exceed* que realiza a gestão de funcionários baseando-se na sua localização (Figura 10).

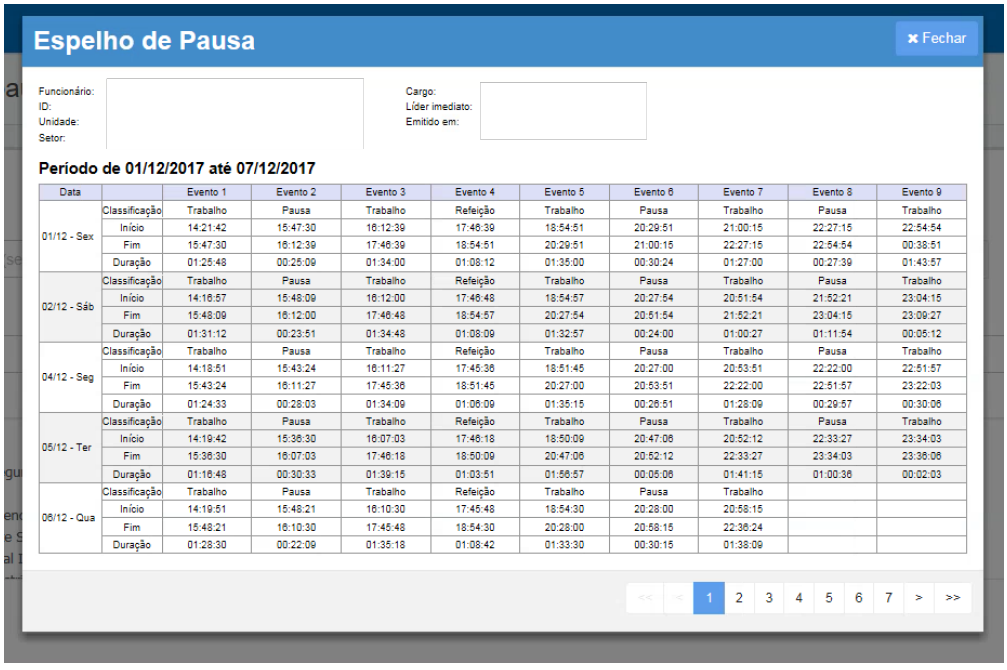

Figura 10: Software SGF

3.2 Sistema de Localização Indoor

Na sequência do trabalho, utilizou-se o *framework* criado no desenvolvimento de uma solução de RTLS para ambientes *indoor* com um nível de precisão sala (*Room-Level Precision*), ou seja, deseja-se estimar em qual sala ou corredor a *tag* encontra-se.

3.2.1 Requisitos do Sistema

Para este sistema, definiu-se que os seguintes requisitos devem ser atendidos:

- baixo custo computacional
- atraso máximo de resposta de 5 segundos
- funcionar em ambientes com divisórias de vidro, madeira e alvenaria

A escolha destes requisitos foi escolhida com o objetivo de se desenvolver um sistema básico, o qual possa ser utilizado em diversos tipos de soluções que façam uso de sistemas RTLS sem a necessidade da alteração do algoritmo de estimativa da localização. Para tanto, o algoritmo deve apresentar um baixo custo computacional, para que seja possível a implementação em servidores mais antigos e/ou não tão potentes, ou para que o sistema possa suportar um grande número de leitores. Além disso, vidro, madeira e alvenaria são as divisórias mais comumente encontradas. Assim, já estará previsto, no sistema, os principais ambientes ao qual o mesmo possa vir a ser instalado. Finalmente, o atraso máximo de resposta de 5 segundos advém da experiencia do autor, o qual, observou que, para tempos maiores que este proposto, a percepção do usuário, referente a localização da *tag*, pode-se tornar desconfortável e/ou confusa.

3.2.2 Materiais

Na implementação do sistema de localização utilizaram-se 5 leitores da marca *Bluecats* modelo *Edge* com antenas omnidirecionais e *tags bluetooth* padrão *Eddystone* (Figura 11), um roteador *Wireless* e um computador.

Figura 11: Leitor Bluecats modelo Edge (Esquerda) e Tags (Direita)

3.2.3 Softwares

Para o desenvolvimento do sistema foram utilizados os seguintes *softwares* e aplicações:

- *Mosquitto*: software que implementa um serviço de *Broker MQTT*
- *Eclipse Neon.3*: *Integrated Development Environment* (IDE) para desenvolvimento em *JAVA*
- *SQL Server*: banco de dados para armazenamento dos resultados da análise

3.2.4 Considerações iniciais

Para realizar-se o sistema proposto será avaliada a intensidade dos sinais entre a *tag* e os diversos leitores. Portanto, quanto maior for o isolamento de sinal entre um ambiente e outro, mais aferido será o sistema de localização. A Tabela 6, adaptada de Holt e Huang 2010, apresenta os valores de perda de sinal para diversos tipos de materiais de construção. Através desta, observa-se que o material que provoca a menor perda na intensidade do sinal, ou o menor isolamento, é o vidro. Sendo assim, o sistema foi desenvolvido em um ambiente cujas divisórias são de vidros simples, o que garantirá que este também funcionará nos demais ambientes.

Tabela 6: Perda de sinal para diferentes materiais

Material ID	Loss (dB)
Tijolo (3,5/7/10,5 cm)	3,5/5/7
Parede de Madeira	8
Porta (Madeira/Metal)	4/12
Vidro (0,25/0,5 cm)	0,8/2
Vidro Blindado	9
Telhado (Seco/Molhado)	5/7
Telhado Plano (Metal)	12
Concreto	12

3.2.5 Instalações e Configurações

Definiu-se que todos os leitor deveria ser instalados o mais próximo possível do centro do cômodo. Assim, em cada sala e corredor, buscou-se encontrar o ponto centrar do mesmo e, então, realizou-se a instalação do leitor no teto, o mais próximo possível deste ponto, de acordo com como a estrutura elétrica permitia. Por fim, instalou-se um roteador *Wireless* em uma posição que fosse capaz de cobrir toda a área do sistema. Essa configuração pode ser vista na planta baixa da Figura 12.

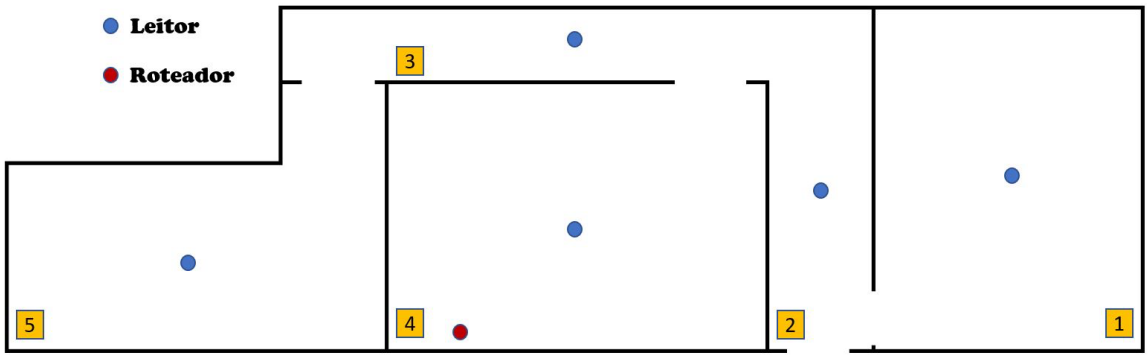

Figura 12: Planta baixa do ambiente e disposição dos equipamentos

Na sequência, instalou-se no servidor os *softwares Mosquitto*, *Eclipse* e *SQL Server*. Por fim, carregou-se o projeto do *framework* na IDE.

3.2.6 Configuração do framework

A fim de utilizar-se o *framework* desenvolvido, realizou-se a configuração dos módulos a serem utilizados neste sistema. Escolheu-se então, utilizar-se um módulo Padronização, dois módulos Transmissão e um módulo Análise. Para módulo Padronização, realizou-se a configuração básica para que o módulo utilizasse os cinco leitores desejados. No primeiro módulo Transmissão (Transmissão 1), selecionou-se o protocolo de comunicação MQTT, configurando-se uma transação de 100 leituras por requisição. No segundo módulo Transmissão (Transmissão 2), selecionou-se o banco de dados *SQL Server*, com uma transação de 1000 leituras por requisição. Por fim, definiu-se que a comunicação entre os módulos Padronização e Análise seriam realizados por intermédio do módulo Transmissão 1, enquanto os eventos de saída do módulo Análise seriam enviados ao módulo Transmissão 2.

3.2.7 Algoritmo

Para que seja realizada a estimativa da localização, a qual uma *tag* encontra-se, desenvolveu-se um sistema de *ranking* que organiza os leitores em ordem do local mais provável para o menos provável, a partir de um valor de *score*. Neste, quanto menor for o valor do *score* de um leitor, melhor será a colocação do mesmo no *ranking*, ou seja, o menor valor será o primeiro colocado e o maior valor será o último. O algoritmo (Figura 13) tem então por objetivo, realizar os passos necessários para a composição do *ranking* a cada execução. Por fim, o mesmo é executado periodicamente a cada 1 segundo e realiza os seguintes passos:

1 - O algoritmo realiza uma requisição de leituras para módulo Transmissão 1, o qual retorna uma lista com as últimas 100 leituras realizadas.

2 - Dessa lista inicial de leituras, são eliminadas todas as com mais de 5 segundos de atraso, em relação ao exato instante, uma vez que deseja-se uma estimativa recente da localização da *tag*.

3 - Após a exclusão, as leituras remanescentes, são separadas em listas diferentes, cada uma contendo apenas as leituras de um único leitor.

4 - Para que um leitor possa participar do *ranking*, o mesmo devo possuir duas ou mais leituras em sua lista. Portanto, o algoritmo elimina todas as listas com um ou zero leituras, a fim de desconsiderar leitores que estejam muito distantes da *tag* em questão, e, por consequência, eliminar a realização de cálculos desnecessários.

5 - Seleciona-se então, as duas leituras mais recentes de cada leitor participante do *ranking*. Dessas leituras, são extraídos os valores de RSSI e então, calculada a média quadrática destes valores. O resultado da média quadrática, de cada leitor, representa o *score* do mesmo.

6 - Por fim, o algoritmo retorna o *ranking* de leitores, já organizada em ordem decrescente de *score*, ou seja, do local mais provável, em que a *tag* esteja, para o menor.

Define-se então, que o primeiro colocado no *ranking* será a localização da *tag*, já que o mesmo tende a ser a estimativa mais confiável.

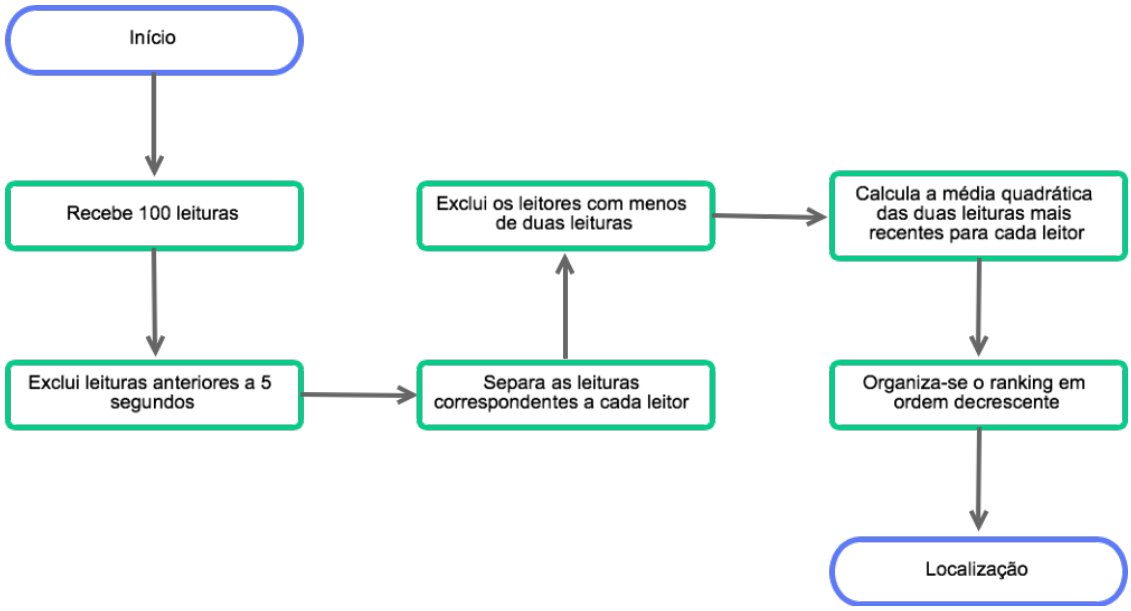

Figura 13: Fluxograma do algoritmo

4 Resultados

Nessa sessão serão apresentados os resultados obtidos a partir da execução a fim de se obter uma estimativa da localização de uma *tag*. Além disso, será realizada uma análise dos métodos utilizados e, por fim uma verificação da qualidade dos resultados obtidos.

4.1 Algoritmo

Com a finalidade de apresentar os resultados intermediários do algoritmo desenvolvido, testou-se o mesmo a partir do solução desenvolvida neste trabalho. Para tanto, uma *tag* foi posicionada na sala do leitor 4. No entanto, buscou-se escolher uma posição que não fosse favorável ao bom funcionamento do sistema. Assim, a mesma foi posicionada em um local que estivesse também próxima a outros leitores para que fosse obtida a maior condição de incerteza possível. Está disposição utilizada pode ser vista na Figura 14.

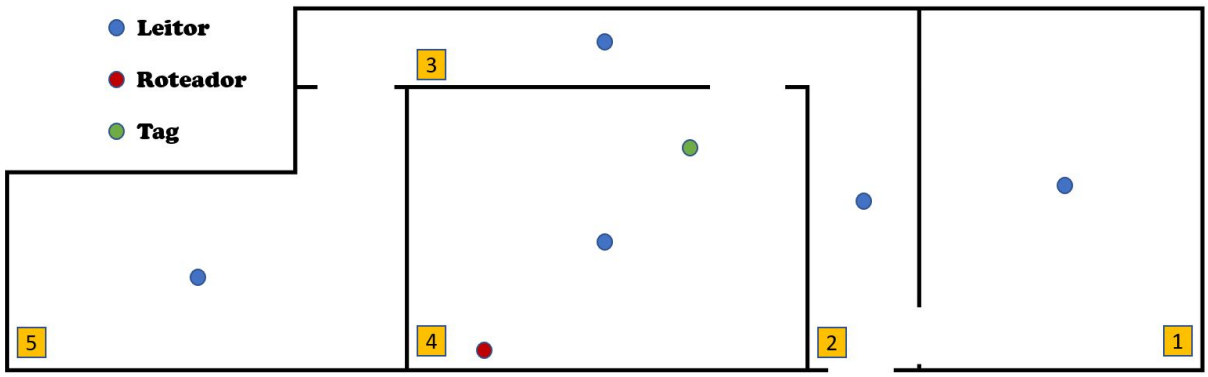

Figura 14: Planta baixa com a disposição dos equipamentos e *tag* do exemplo apresentado

Após requisitar e receber as leituras do módulo Transmissão 1, o algoritmo elimina as leituras recebidas com tempos anteriores a 5 segundos. Com esta exclusão, a lista de leituras reduziu-se de 100 a 31 leituras, como pode-se observar na Tabela 7.

Tabela 7: Leituras com tempos inferiores a 5 segundos

Scanner MAC	RSSI (dB)	Timestamp (ms)	Time delay (ms)
E4956E4E40A3	-73	1512161650504	4892
E4956E4E40F2	-73	1512161650519	4877
E4956E4E411F	-79	1512161650522	4874
E4956E4E40A3	-73	1512161651016	4380
E4956E4E40F2	-73	1512161651027	4369
E4956E4E411F	-79	1512161651031	4365
E4956E4E40F2	-72	1512161651524	3872
E4956E4E411F	-79	1512161651525	3871
E4956E4E40A3	-73	1512161652014	3382
E4956E4E40F2	-73	1512161652031	3365
E4956E4E411F	-80	1512161652033	3363
E4956E4E40A3	-73	1512161652519	2877
E4956E4E40F2	-73	1512161652536	2860
E4956E4E411F	-78	1512161652538	2858
E4956E4E40A3	-73	1512161653022	2374
E4956E4E40F2	-73	1512161653037	2359
E4956E4E411F	-79	1512161653039	2357
E4956E4E40A3	-68	1512161653513	1883
E4956E4E4068	-93	1512161653524	1872
E4956E4E40F2	-73	1512161653530	1866
E4956E4E411F	-78	1512161653534	1862
E4956E4E4068	-93	1512161654031	1365
E4956E4E40F2	-77	1512161654037	1359
E4956E4E411F	-78	1512161654041	1355
E4956E4E40A3	-68	1512161654341	1055
E4956E4E40A3	-68	1512161654518	878
E4956E4E4068	-92	1512161654530	866
E4956E4E40F2	-77	1512161654535	861
E4956E4E411F	-78	1512161654540	856
E4956E4E40A3	-67	1512161655021	375
E4956E4E4068	-93	1512161655032	364

Na sequência, é necessário que as leituras remanescentes sejam separadas por leitores em listas separadas. Para tanto, o algoritmo realiza este procedimento enquanto conta a quantidade de leituras referentes a cada leitor. A Tabela 8 apresenta a relação dos leitores utilizados e o número de leituras obtidas por cada um.

Tabela 8: Contagem de leituras por leitor

Scanner ID	Scanner MAC	Número de Leituras
1	E4956E4E411F	9
2	E4956E4E4068	4
3	E4956E4E40F2	9
4	E4956E4E40A3	9
5	E4956E4E405C	0

A partir dos resultados desta contagem, o algoritmo exclui o leitor 5 já que o mesmo não possui o mínimo de leituras necessárias. Em seguida, são selecionadas as últimas duas leituras mais recentes para cada um dos leitores que estará presente no *ranking* (Nesta situação, Leitores 1, 2, 3 e 4). As Tabelas 9, 10, 11 e 12 apresentam uma relação destas leituras para cada um dos leitores.

Tabela 9: Seleção das últimas duas leituras mais recentes do leitor 1

RSSI (dB)	Timestamp (ms)	Time delay (ms)
-78	1512161654041	1355
-78	1512161654540	856

Tabela 10: Seleção das últimas duas leituras mais recentes do leitor 2

RSSI (dB)	Timestamp (ms)	Time delay (ms)
-92	1512161654530	866
-93	1512161655032	364

Tabela 11: Seleção das últimas duas leituras mais recentes do leitor 3

RSSI (dB)	Timestamp (ms)	Time delay (ms)
-77	1512161654037	1359
-77	1512161654535	861

Tabela 12: Seleção das últimas duas leituras mais recentes do leitor 4

RSSI (dB)	Timestamp (ms)	Time delay (ms)
-68	1512161654518	878
-67	1512161655021	375

Por fim, é calculado o *score* para cada leitor e organizado uma lista com o *ranking* final dos leitores. Vale ressaltar que quanto menor for o valor do *score* para um leitor, melhor este estará a sua posição. A Tabela 13 apresenta os resultados do *ranking* final dos leitores para este teste. Observa-se que o primeiro colocado, o qual representa a melhor estimativa da posição da *tag*, é o leitor de número 4. O que é um resultado satisfatório, já que a *tag* realmente encontra-se na sala do leitor 4.

Tabela 13: Ranking final dos leitores

RANK	ID	SCANNER	SCORE
1º	4	E4956E4E40A3	4556.5
2º	3	E4956E4E40F2	5929
3º	1	E4956E4E411F	6084
4º	2	E4956E4E4068	8556.5

4.2 Filtros

Muitas vezes, o sinal de RSSI de um *beacon bluetooth* pode apresentar perturbações de alta frequência. Para tentar-se resolver este problema, foram realizados teste utilizando-se filtros *Finite Impulse Response* (FIR) e filtros de Kalmann. No entanto, a utilização dos mesmos foi descartada devido ao grande atraso ao qual estes filtros causavam na dinâmica do sistema. Em busca de uma solução para a resolução deste problema, observou-se que, o sinal recebido, oscilava alternadamente de um valor mais alto para um valor mais baixo e, em seguida, novamente para um valor mais alto. Percebeu-se então, que a realização da média entre os últimos dois valores resultaria em uma redução da perturbação em questão. Este resultado pode ser observado na Figura 15.

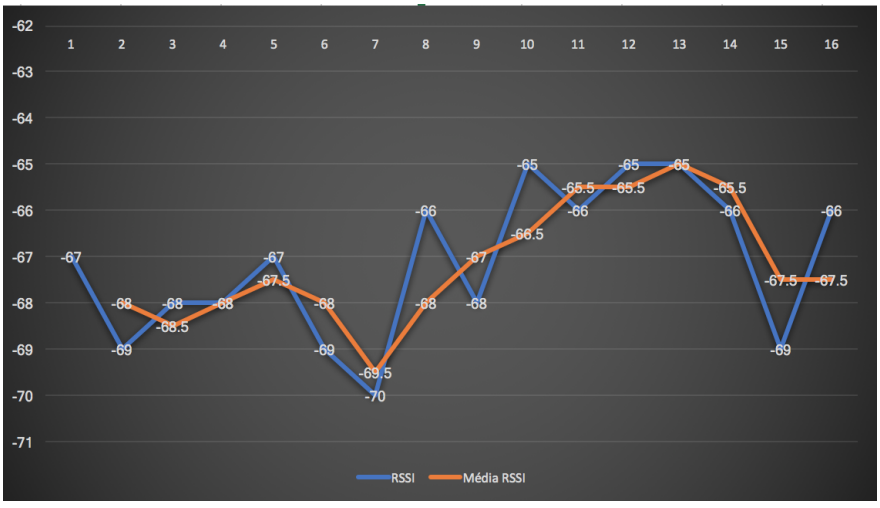

Figura 15: Sinal de RSSI lido pelo leitor e sinal calculado a partir da média dos últimos dois valores

4.3 Média Quadrática

Outra dificuldade, presente no desenvolvimento deste sistema, foi a realização da diferenciação dos valores muito próximos de RSSI entre os leitores, para que fosse possível determinar-se uma estimativa de posição. Como o objetivo deste sistema era buscar uma localização com precisão de nível sala (*Room-Level Precision*) técnicas como as de triangulação e trilateralização foram descartadas, já que as mesmas estimam coordenadas de posição local desnecessárias para a precisão desejada. Realizou-se então, testes calculando-se o quadrado dos valores de RSSI, a fim de tornar mais evidente as diferenças entre os valores muito próximos. Estes testes, apresentaram resultados bons e ruins. A partir do cálculo do quadrado dos valores, foi possível ressaltar a diferença entre os valores de um leitor e outro. No entanto, a instabilidade da intensidade do sinal bruto fazia com que a estimativa de posição realizasse saltos periódicos entre leitores vizinhos. Uniu-se então, a solução do problema de filtragem as conclusões obtidas nos testes realizados e realizou-se um novo teste utilizando-se o cálculo da Média Quadrática dos últimos dois valores. Resultados satisfatórios foram então obtidos, tornando-se possível a realização de estimativas de boa qualidade utilizando-se este método.

4.4 Verificação da qualidade das estimativas

A fim de verificar-se a qualidade dos resultados obtidos, foram espalhadas 120 *tags* pelo espaço ao qual o sistema estava instalado e realizou-se 1000 estimativas de localização para cada uma. Verificou-se então que, em casos nos quais a *tag* localizava-se próxima à fronteira de um ambiente com o outro, erros de leitura eram mais frequentes, pois leitores vizinhos eram favorecidos em relação ao leitor onde efetivamente encontrava-se a *tag*. Além disso, em momentos aleatórios, leitores fora das vizinhanças do leitor em questão acabavam sendo escolhidos como a melhor estimativa. a partir destes resultados, definiu-se que a qualidade de acerto era de $72,5\%$, utilizando-se como base o pior resultado. A Figura 16 apresenta os resultados obtidos no teste realizado.

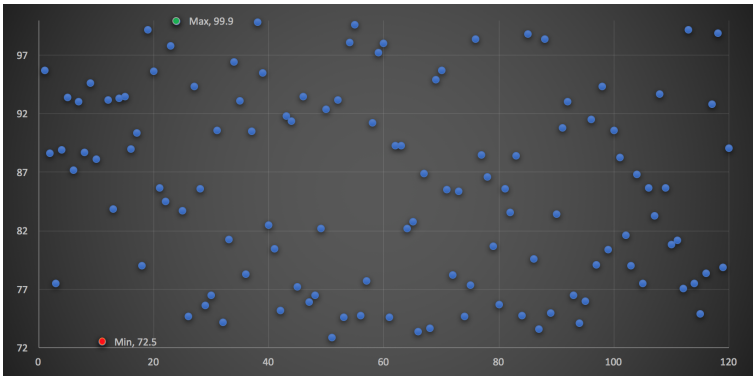

Figura 16: Qualidade de acertos para *tags* com potência de sinal de -20 dB

Em busca de uma solução para este problema, reduziu-se a potência do sinal de transmissão, das *tags*, do padrão de fabrica de -20 dB para um valor de -30 dB. Realizou-se novamente o teste realizado anteriormente, e uma grande melhora nos resultados pode ser observada. Com a redução da intensidade do sinal, a taxa de acertos do sistema subiu para $96,3\%$ no pior dos casos. Além disso, as estimativas erradas foram todas estimadas em leitores vizinhos ao leitor em questão. Estes resultados foram considerados aceitáveis para este sistema, já que para um período de 1 hora, um total de apenas 1,67 minutos estariam errados. Mais além, este tempo em que as estimativas estão incorretas não é contínuo, mas sim fracionado por todo o período, fazendo-se com que seu efeito seja mais ainda minimizado, além de poder ser suprimido com uma rotina de pós-processamento. A Figura 17 apresenta os resultados obtidos neste novo teste realizado.

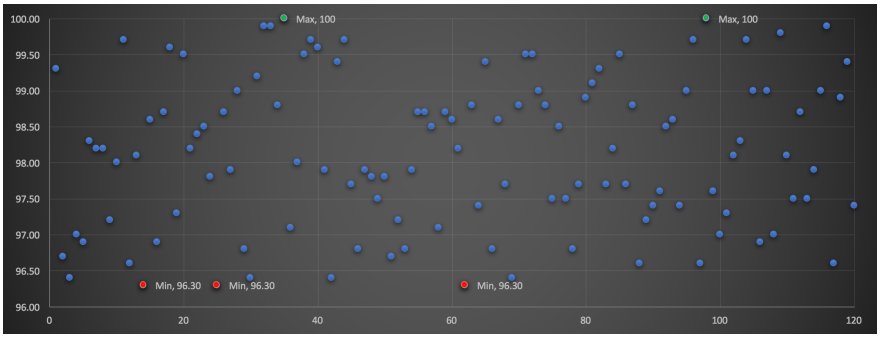

Figura 17: Qualidade de acertos para *tags* com potência de sinal de -30 dB

5 Conclusões

Através da realização deste trabalho pode-se concluir que o sistema de localização *indoor* desenvolvido bem como o *framework* criado funcionam e satisfazem os requisitos propostos. A performance do sistema final projetado garante $96,3\%$ de acerto na identificação da localização correta da *tag*.

A metodologia aplicada através do *framework* gerou mais praticidade e facilidade de integração e utilização do sistema como um todo. A estrutura modular obtida tornou o sistema facilmente escalável, ou seja, ele suporta sistemas de pequeno, médio e grande porte.

A abstração gerada pelo *framework* permite a adição de novos modelos de leitores e também o uso de novos softwares e estruturas de armazenamento de dados, uma vez que criou classes e objetos genéricos. Sendo assim, ao adicionar-se um novo dispositivo ou estrutura de armazenamento é necessário somente acrescentar um novo suporte para este no *framework*.

Devido à flexibilidade introduzida no sistema, este passou a suportar integração multi-plataforma, ou seja, é possível intercambiar dados entre estruturas de armazenamento diferentes de uma forma prática.

A partir dos testes realizados, pode-se concluir que o sistema apresentou resultados satisfatórios para o procedimento utilizado, já que , os mesmos, alcançaram uma taxa de acerto considerada aceitável para o projeto desenvolvido. O mesmo, baseando na média quadrática das últimas duas leituras, tornou o uso de filtros dispensável. Isso deve-se ao fato de que, o uso destes, além de provocar um atraso na resposta, não traria uma melhora significativa para o sistema.

Por fim, os próximos passos no desenvolvimento deste trabalho seriam a automatização do processo de criação de um novo projeto, a implementação no *framework* da comunicação no sentido algoritmo-tag e a otimização da inserção de dados nas estruturas de armazenamento. Além disso, é relevante o desenvolvimento de uma análise do limite da capacidade de leitura dos leitores (número de *tags* suportadas) e limite de *inputs* por segundo nas estruturas de armazenamento (bancos de dados).

Referências

Betters, Elyse e Chris Hall (2017). *What is ZigBee and why is it important for your smart home?* Ed. por Pocket-lint. URL: https://www.pocket-lint.com/smart-home/news/129857-what-is-zigbee-and-why-is-it-important-for-your-smart-home (acesso em 28/12/2017).

Boulos, Maged N Kamel e Geoff Berry (2012). "Real-time locating systems (RTLS) in healthcare: a condensed primer". Em: *International journal of health geographics* 11.1, p. 25.

Casaroli, M. (2017). *Os motes da Internet das Coisas*. Ed. por Embarcados. URL: https://www.embarcados.com.br/os-motes-da-internet-das-coisas/ (acesso em 07/12/2017).

Castro, A. P. P. de (2017). Ed. por O que RFID. URL: https://www.gta.ufrj.br/grad/07_1/rfid/RFID_arquivos/o%5C%20que%5C%20e.html (acesso em 07/12/2017).

CCM, ed. (2017). *Como funciona o Wi-Fi*. URL: http://br.ccm.net/contents/790-como-funciona-o-wi-fi (acesso em 28/12/2017).

Decreto-lei nº 5.452, Artigo nº 253 (1943). Plenário. Relator: Alexandre Marcondes Filho. Brasil. Casa Civil, Presidência da República. Brasília.

Heydon, Robin (2013). *Bluetooth low energy: the developer's handbook*. Vol. 1. Prentice Hall Upper Saddle River.

Holt, Alan e Chi-Yu Huang (2010). *802.11 wireless networks: security and analysis*. Springer Science & Business Media.

"IEEE 802.11" (2016). Em: *IEEE Std 802.11-2016 (Revision of IEEE Std 802.11-2012)*, pp. 1–3534.

Li, Shuai et al. (2013). "Neural network based mobile phone localization using bluetooth connectivity". Em: *Neural Computing and Applications* 23.3-4, pp. 667–675.

Malik, A. (2009). *RTLS for Dummies*. John Wiley & Sons.

Morimoto, C. E. (2017). *UWB*. Ed. por Guia do Hardware. URL: http://www.hardware.com.br/termos/uwb (acesso em 07/12/2017).

Nunes, B. (2017). *Introdução a LoRa®, NB-IoT e Sigfox*. Ed. por Embarcados - Sua fonte de informações sobre Sistemas Embarcados. URL: https://www.embarcados.com.br/lora-nb-iot-e-sigfox/ (acesso em 07/12/2017).

Oracle, ed. (2017). *Java Overview*. URL: https://www.oracle.com/java/index.html (acesso em 28/12/2017).

Rida, Mohamed Er et al. (2015). "Indoor location position based on bluetooth signal strength". Em: *Information Science and Control Engineering (ICISCE), 2015 2nd International Conference on*. IEEE, pp. 769–773.

Somensi, R. M. (2016). "Carga de trabalho do enfermeiro: comparação entre método observacional e on-line".

Tigoe (2017). *tigoe/BLEDocs*. Ed. por GitHub. URL: https://github.com/tigoe/BLEDocs/wiki/Introduction-to-Bluetooth-LE (acesso em 07/12/2017).

TomTom, ed. (2017). *O QUE É O GPS?* URL: http://br.support.tomtom.com/app/answers/detail/a_id/18311/~/o-que-%5C%C3%5C%A9-o-gps%5C%253F (acesso em 07/12/2017).

Vicenzi, Alexandre (2017). *O que é MQTT?* Ed. por Buteco Open Source. URL: http://www.butecopensource.org/mqtt-parte-1-o-que-e-mqtt/ (acesso em 07/12/2017).