

**UNIVERSIDADE FEDERAL DO MARANHÃO
CENTRO DE CIÊNCIAS EXATAS E TECNOLÓGICAS
CURSO DE CIÊNCIA DA COMPUTAÇÃO**

DANIEL SOARES CARVALHO

**UMA ABORDAGEM DE IOT PARA GESTÃO DA PRESENÇA E ENCONTROS DE
PESSOAS EM ESPAÇOS FÍSICOS**

**São Luís
2020**



DANIEL SOARES CARVALHO

**UMA ABORDAGEM DE IOT PARA GESTÃO DA PRESENÇA E ENCONTROS DE
PESSOAS EM ESPAÇOS FÍSICOS**

Monografia apresentada ao curso de
Ciência da Computação da Universidade
Federal do Maranhão, como parte dos
requisitos necessários para obtenção
do grau de Bacharel em Ciência da
Computação. Orientador: Prof. Dr.
Francisco José da Silva e Silva

São Luís
2020

Ficha gerada por meio do SIGAA/Biblioteca com dados fornecidos pelo(a) autor(a).
Núcleo Integrado de Bibliotecas/UFMA

Carvalho, Daniel Soares.

Uma abordagem de IoT para gestão da presença e encontros de pessoas em espaços físicos / Daniel Soares Carvalho. - 2020.

43 f.

Orientador(a): Francisco José da Silva e Silva.

Monografia (Graduação) - Curso de Ciência da Computação, Universidade Federal do Maranhão, São Luís, 2020.

1. Beacon BLE. 2. Gestão de Encontros. 3. Gestão de Presença. 4. Internet das Coisas. I. Silva, Francisco José da Silva e. II. Título.

DANIEL SOARES CARVALHO

**UMA ABORDAGEM DE IOT PARA GESTÃO DA PRESENÇA E ENCONTROS DE
PESSOAS EM ESPAÇOS FÍSICOS**

Monografia apresentada ao curso de
Ciência da Computação da Universidade
Federal do Maranhão, como parte dos
requisitos necessários para obtenção
do grau de Bacharel em Ciência da
Computação. Orientador: Prof. Dr.
Francisco José da Silva e Silva

Aprovada em 10/01/2020

**Prof. Dr. Francisco José da Silva e
Silva** (Orientador)
Universidade Federal do Maranhão

**Prof. Dr. Alexandre César Muniz de
Oliveira**
Universidade Federal do Maranhão

Prof. Dr. Geraldo Braz Junior
Universidade Federal do Maranhão

São Luís
2020

Agradecimentos

Agradeço em primeiro lugar à Deus, porque Dele e por Ele, e para Ele, são todas as coisas.

Agradeço a minha família, por todo amor e carinho, me apoiando mesmo quando decidi mudar para o curso de Ciência da Computação.

Agradeço ao meu orientador, Prof. Dr. Francisco Silva, pela oportunidade de trabalhar em um projeto com um tema que me identifico, também pelo auxílio durante a escrita deste trabalho, sendo paciente mesmo quando eu precisava dele em momentos inoportunos ou para algo feito às pressas.

Agradeço a todos os professores do Departamento de Informática, que me auxiliaram na minha trajetória. Principalmente à Prof. Dra. Simara da Rocha, que sempre teve paciência comigo, tanto como professora de disciplina, quanto como coordenadora do curso. Escusado será dizer que talvez eu nem conseguiria me formar se não fosse sua dedicação a seus alunos.

Em especial, agradeço aos Prof. Dr. Geraldo Braz e Prof. Dr. Alexandre César, por terem me auxiliado em diversos momentos durante todo o curso, e por fim ainda terem aceito participar da banca de defesa. Não apenas os tenho como professores, mas como amigos.

Por fim, agradeço a todos os amigos e colegas que conheci durante meu tempo na UFMA, pois eles foram de vital importância para meu crescimento como aluno, profissional e pessoa.

*“Aqueles que se sentem
satisfeitos sentam-se e nada fazem.
Os insatisfeitos são os únicos
benfeitores do mundo.”*

(Walter S. Landor)

Resumo

Desde o começo da difusão do *Bluetooth* de baixa energia (*Bluetooth Low Energy*) em 2011, essa tecnologia marcada pelo baixo consumo de energia, baixo preço e transmissão de mensagens pequenas, impulsionou pesquisas de IoT voltadas à localização de pessoas, associando dispositivos emissores de sinais BLE à dispositivos móveis. No entanto, os sistemas e aplicativos desenvolvidos, geralmente eram criados utilizando requisitos específicos de cada cenário, uma sala, um condomínio, uma empresa. Para se criar uma aplicação que localizasse uma pessoa em um espaço, era necessário criar todo o sistema do zero, focando apenas no contexto específico. O desenvolvimento deste trabalho possui como objetivo a construção de um serviço capaz de prover informações para gestão da presença e encontros de pessoas em espaços físicos genéricos através de dados coletados por dispositivos de IoT. Para o desenvolvimento deste trabalho são utilizados dispositivos chamados *beacons*, que emitem sinais de rádio frequência em intervalos de tempo pré-definidos, smartphones e um aplicativo Android que encontra dispositivos e publica encontros. A base para o desenvolvimento do trabalho é o HORYS, software que gerencia dados de encontros entre dispositivos, e o Servidor Semântico, que associa dispositivos aos seus portadores. Como resultados, foi concluído que a proposta do projeto foi atingida, obtendo-se os resultados das buscas definidas de forma eficaz, e constatado-se que o impacto causado pelo serviço proposto impacta minimamente no tempo de retorno do sistema ao usuário final.

Palavras-chave: *Beacons* BLE, Gestão de Presença, Gestão de Encontros, Internet das Coisas;

Abstract

Since the start of the broadcast of low-power *Bluetooth (Bluetooth Low Energy)* in 2011, this technology, marked by low power consumption, low price and small message transmission, boosted searches for IoT aimed at locating people by associating BLE signal-emitting devices to mobile devices. However, these systems and applications developed were usually created using specific requirements of each scenario, a room, a condominium, a company. To create an application that located a person in a space, it was necessary to create the whole system from scratch, focusing only on a specific context. The development of this work aims at the construction of a service capable of providing information for presence and people rendezvous management in a general physical spaces through data collected by IoT devices. For the development of this work we used devices called *beacons*, which emit radio frequency signals at predefined time intervals, smartphones and an Android application that finds devices and publishes *rendezvous*. The basis for the development of the work will be HORYS, a software that manages data of encounters between devices, and a Semantic Server, that associates devices to its holders. As a result, it was concluded that the project proposal was reached, obtaining effectively the results of the defined searches, and it was found that the impact caused by the proposed service has minimal impact on the system's return time to the end user.

Keywords: *Beacons* BLE, Presence Management, Rendezvous Management, Internet of Things;

Lista de ilustrações

Figura 1 – Funcionamento de um <i>beacon</i>	18
Figura 2 – Os diferentes IDs de um anúncio <i>iBeacon</i>	18
Figura 3 – Quando um dispositivo está mais distante do <i>beacon</i> , a intensidade do sinal é diminuída e, portanto, a estimativa de precisão aumentará.	19
Figura 4 – A intensidade do sinal aumenta à medida que um dispositivo se aproxima do <i>beacon</i> , levando a uma melhor estimativa de proximidade.	19
Figura 5 – Arquitetura proposta do projeto.	22
Figura 6 – Estrutura de dados utilizada pelo HORYS.	23
Figura 7 – Exemplo de JSON de um <i>rendezvous</i> enviado para o HORYS.	23
Figura 8 – Diagrama de Entidade Relacionamento do Servidor Semântico.	24
Figura 9 – Fluxograma de um caso de uso específico.	31
Figura 10 – Diagrama de pacotes.	33
Figura 11 – Diagrama de classes do pacote <i>core</i>	33
Figura 12 – Diagrama de classes do pacote <i>database</i>	34
Figura 13 – Diagrama de classes do pacote <i>data</i>	34

Lista de tabelas

Tabela 1 – Os requisitos funcionais identificados.	25
Tabela 2 – Os requisitos não funcionais identificados.	25
Tabela 3 – Os requisitos funcionais identificados.	26
Tabela 4 – Os requisitos não funcionais identificados.	27
Tabela 5 – Experimento 1 realizado às 14h50min	37
Tabela 6 – Experimento 1 realizado às 15h10min	38
Tabela 7 – Experimento 1 realizado às 15h30min	38
Tabela 8 – Experimento 2	39

Lista de abreviaturas e siglas

UFMA	Universidade Federal do Maranhão
IoT	<i>Internet of Things</i>
IoMT	<i>Internet of Mobile Things</i>
RFID	<i>Radio-Frequency IDentification</i>
LSDi	Laboratório de Sistemas Distribuídos Inteligentes
LAC	<i>Laboratory for Advanced Collaboration</i>
PUC-Rio	Pontifícia Universidade Católica do Rio de Janeiro
BLE	<i>Bluetooth Low Energy</i>
RSSI	<i>Received Signal Strength Indication</i>
HORYS	<i>Hub-Object Rendezvous Registry Service</i>
REST	<i>REpresentational State Transfer</i>
JSON	<i>JavaScript Object Notation</i>
SOAP	<i>Simple Object Access Protocol</i>
WSDL	<i>Web Services Description Language</i>
SOA	<i>Service Oriented Architecture</i>
HTTP	<i>HyperText Transfer Protocol</i>
URL	<i>Uniform Resource Locator</i>
URI	<i>Uniform Resource Identifier</i>
XML	<i>eXtensible Markup Language</i>

Sumário

	Lista de tabelas	9
1	INTRODUÇÃO	13
1.1	Contextualização	13
1.2	Objetivos do Trabalho	13
1.2.1	Objetivos	13
1.2.2	Perguntas Pertinentes	14
1.3	Organização da Monografia	14
2	FUNDAMENTAÇÃO TEÓRICA	15
2.1	IoT	15
2.2	IoMT	16
2.3	Bluetooth	17
2.3.1	Bluetooth Low Energy	17
2.4	Beacons e iBeacons	17
2.5	M-Hub	18
2.6	Arquitetura Orientada a Serviços	19
2.6.1	Serviços Web	20
2.6.1.1	Serviços Web RESTful	20
3	SISTEMA DE GESTÃO DE PRESENÇA	21
3.1	Contextualização do Sistema	21
3.2	Justificativa	21
3.3	Objetivos do Sistema	21
3.4	Arquitetura do Sistema	22
3.4.1	Visão Geral do Componentes	22
3.4.2	HORYS	23
3.4.3	Servidor Semântico	24
3.5	Requisitos do Sistema de Gestão de Presença	24
3.5.1	Requisitos Funcionais	25
3.5.2	Requisitos Não Funcionais	25
4	SERVIÇO ATENDENTE	26
4.1	Especificação dos Requisitos	26
4.1.1	Requisitos Funcionais	26
4.1.2	Requisitos Não Funcionais	27
4.2	Arquitetura do Serviço	27
4.3	Interface	27
4.3.1	Exemplo com Diagrama de Atividade	30
4.4	Aspectos de Implementação	32
4.4.1	Ferramentas e Tecnologias	32
4.4.2	Implementação	32

5	AVALIAÇÃO	35
5.1	Experimento 1	35
5.1.1	Métricas	35
5.1.2	Cenário	35
5.1.3	Simulação de Locomoção na Residência	36
5.1.4	Simulação dos <i>Beacons</i>	37
5.1.5	Simulação do Sistema de Gestão de Presença	37
5.1.6	Recursos computacionais	37
5.1.7	Resultados	37
5.1.8	Análise dos Resultados	38
5.2	Experimento 2	38
5.2.1	Simulação do Sistema de Gestão de Presença	38
5.2.2	Resultados	39
5.2.3	Análise dos Resultados	39
5.3	Análise Final	39
6	CONCLUSÃO	40
7	TRABALHOS FUTUROS	41
	REFERÊNCIAS	42

1 INTRODUÇÃO

1.1 Contextualização

A *Internet das Coisas* (IoT - *Internet of Things*) é uma extensão da Internet atual, que proporciona aos objetos do dia a dia, com capacidade computacional e de comunicação, se conectarem à Internet. Essa conexão viabiliza o acesso e controle remotos à objetos e ainda pode permitir que os próprios objetos sejam acessados como provedores de serviços. Usando os recursos desses objetos será possível detectar seu contexto, controlá-lo, viabilizar troca de informações uns com os outros, acessar serviços da Internet e interagir com pessoas. Com isso, uma gama de novas possibilidades de aplicações surgem (ex: cidades inteligentes (*Smart Cities*), saúde (*Healthcare*), casas inteligentes (*Smart Home*)) (SANTOS et al., 2016).

Segundo Mattern e Floerkemeier (2010), objetos “inteligentes” desempenham um papel fundamental na *Internet das Coisas*, uma vez que possuem a habilidade de se comunicar. Usando sensores, essas “coisas” (do inglês, *things*) são capazes de perceber seu ambiente e, por meios de diversas técnicas de comunicação, podem se comunicar umas com as outras, prover e usar serviços, prover dados e reagir a eventos.

Temos também a *Internet das Coisas Móveis* (IoMT - *Internet of Mobile Things*), uma derivação da IoT padrão, focada em considerar a mobilidade das *things*, proporcionada pelo fato de smartphones e carros estarem vindo com sensores cada vez mais avançados. Essa consideração com a mobilidade do dispositivo, apresenta novas considerações a serem estudadas, como por exemplo onde está localizado atualmente o dispositivo móvel, nas mãos de quem (NAHRSTEDT et al., 2016).

No contexto de IoT, podemos chamar de *smart objects*, objetos do dia a dia equipados com componentes de capacidade computacional para processamento de tarefas, de comunicação para se conectar com outros objetos ou sensores e atuadores para conhecer o ambiente ao seu redor e o controlar (FORTINO et al., 2014).

1.2 Objetivos do Trabalho

As tecnologias de localização baseadas em IoT possuem um grande número de possibilidades e áreas de uso. Nesta seção é descrito os objetivos e cenários motivacionais que incentivaram este projeto.

1.2.1 Objetivos

O objetivo deste trabalho de conclusão de curso é propor um serviço para prover informações necessárias para a gestão de presenças e encontros de pessoas em espaços físicos através de dados coletados a partir de um dispositivo de IoT

chamado *beacon* BLE. No contexto deste trabalho, a gestão de presenças e encontros de pessoas refere-se a identificação de informações relativas à presença física de indivíduos em espaços físicos (como um laboratório, uma sala de aula, ou o consultório de um médico) bem como a co-localização de indivíduos nestes espaços físicos visando ou não a realização de tarefas coletivas. Neste trabalho é considerado também o contexto temporal (quando e por quanto tempo elas estiveram no espaço físico).

1.2.2 Perguntas Pertinentes

O software desenvolvido oferece uma interface através da qual pode-se obter informações como:

- a) Quem está em um determinado espaço físico neste momento?
- b) Quem esteve em um determinado espaço físico em um tempo específico?
- c) Por quanto tempo esta pessoa ficou nesse espaço físico?
- d) Essa pessoa se encontrou com outras pessoas em um dado momento?

1.3 Organização da Monografia

Este trabalho está organizado em cinco capítulos, de forma a apresentar o conteúdo mais claramente, conforme os parágrafos a seguir.

O Capítulo 2, é apresentada a fundamentação teórica necessária para compreensão e embasamento deste estudo.

O Capítulo 3, neste capítulo é apresentado o Sistema de Gestão de Presença utilizado como base para este trabalho.

O Capítulo 4, neste capítulo é descrito o serviço proposto por este trabalho.

O Capítulo 5, neste capítulo é apresentados e discutidos os resultados obtidos neste trabalho.

O Capítulo 6, neste capítulo é apresentada a conclusão e os trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo apresenta a fundamentação teórica utilizada no desenvolvimento deste trabalho que é necessária para compreensão das técnicas utilizadas na metodologia proposta para prover informações necessárias para a gestão de presenças e encontros de pessoas em espaços físicos. Assim, ao decorrer do capítulo será discutida a base para o desenvolvimento deste estudo, partindo da compreensão do que se trata a *Internet das Coisas* e a tecnologia *Bluetooth*.

2.1 IoT

A *Internet das Coisas* (IoT) é um paradigma recente que está rapidamente ganhando terreno no cenário das modernas telecomunicações sem fio. A ideia básica deste conceito é a presença generalizada em torno de nós, de uma variedade de coisas ou objetos - tais como identificadores de Identificação por Radiofrequência (RFID - *Radio Frequency Identification*), sensores, atuadores, telefones celulares, etc. - que, através de protocolos de comunicação, são capazes interagir uns com os outros e cooperar com seus vizinhos para alcançar objetivos comuns (ATZORI; IERA; MORABITO, 2010).

A IoT pode ser vista como a combinação de diversas tecnologias, de acordo com Santos et al. (2016), ela pode ser dividida em blocos básicos, sendo eles:

a) Identificação:

- tecnologias como RFID, NFC (*Near Field Communication*) e endereçamento IP podem ser empregados para identificar os objetos.

b) Sensores/Atuadores:

- sensores coletam informações sobre o contexto onde os objetos se encontram;
- atuadores podem manipular o ambiente ou reagir de acordo com os dados lidos.

c) Comunicação:

- diz respeito às diversas técnicas usadas para conectar objetos inteligentes;
- algumas das tecnologias usadas são WiFi, Bluetooth, IEEE 802.15.4 e RFID.

d) Computação:

- responsáveis por executar algoritmos locais nos objetos inteligentes;
- por exemplo, microcontroladores, processadores e FPGAs.

e) Serviços, destacando-se:

- Serviços de Identificação, responsáveis por mapear Entidades Físicas (EF) (de interesse do usuário) em Entidades Virtuais (EV) como, por exemplo, a temperatura de um local físico em seu valor, coordenadas geográficas do sensor e instante da coleta;
- Serviços de Agregação de Dados que coletam e sumarizam dados homogêneos/heterogêneos obtidos dos objetos inteligentes;
- Serviços de Colaboração e Inteligência que agem sobre os serviços de agregação de dados para tomar decisões e reagir de modo adequado a um determinado cenário.

f) Semântica:

- Trata da descoberta de conhecimento e uso eficiente dos recursos existentes na IoT, a partir dos dados existentes, com o objetivo de prover determinado serviço;
- podem ser usadas diversas técnicas como *Resource Description Framework* (RDF), *Web Ontology Language* (OWL) e *Efficient XML Interchange* (EXI).

2.2 IoMT

A IoT geralmente lida com objetos estáticos, frequentemente presentes na estrutura do ambiente, como sensores fazendo a medição da temperatura de um objeto ou local, trancas elétricas para desbloquear portas na presença de leitura de um RFID, etc. Em adição à isso, temos a *Internet das Coisas Móveis* (IoMT), um ramo da IoT focado em *smart objects* com capacidade de mobilidade, como por exemplo, smartphones, chaveiros com RFID ou luvas equipadas com sensores táteis.

De acordo com Nahrstedt et al. (2016), a IoMT se diferencia da IoT nos seguintes aspectos:

a) *contexto*, exemplo:

- onde e com quem o dispositivo móvel se encontra.

b) *acesso à Internet e conectividade*, exemplo:

- estado de conexão (conectado/desconectado);
- se conectado, em qual rede.

c) *disponibilidade de energia*, exemplo:

- onde o dispositivo pode ser recarregado;
- quanta energia o aplicativo necessita.

d) *segurança e privacidade*, exemplo:

- que tipo de infraestrutura de segurança o dispositivo encontra ao mudar de localização.

2.3 Bluetooth

O *Bluetooth* é um protocolo de comunicação proposto pela Ericsson para substituir a comunicação serial RS-232 em 1994 (ARFWEDSON; SNEDDON, 1999). Atualmente o *Bluetooth Special Interest Group* é responsável por criar, testar e manter essa tecnologia. Além disso, o *Bluetooth* é uma das principais tecnologias de rede sem fio para PANs – *Personal Area Networks*, que é utilizada em smartphones, headsets, PCs e outros dispositivos (SANTOS et al., 2016). Entretanto, a versão padrão focou em aumentar a taxa de comunicação, tornando o protocolo mais complexo e, por consequência, não otimizado para dispositivos com limitações energéticas.

2.3.1 Bluetooth Low Energy

Em 2010, foi desenvolvido pelo *Bluetooth Special Interest Group*, uma nova tecnologia, chamada de *Bluetooth Low Energy* (BLE), também comercializada como *Bluetooth Smart*, que foi integrada à versão 4.0 da *Bluetooth Core Specification*, as especificações da tecnologia *Bluetooth* (SANTOS et al., 2016).

Como uma extensão do *Bluetooth*, o BLE mantém o mesmo alcance do clássico mas com um consumo de energia reduzido. Essa nova tecnologia também permite a possibilidade de se comunicar sem a necessidade de emparelhar dispositivos. Com o BLE, surgiu uma nova área de uso na forma de *beacons*, abrindo as portas para um grande conjunto de novos casos de uso (ANTON; RUNBERT, 2015).

2.4 Beacons e iBeacons

Beacons são dispositivos que transmitem sinais via BLE para todos os dispositivos dentro do seu alcance. O *Beacon* usa o BLE para transmitir um sinal e pode ser usado para determinar se um dispositivo está dentro de uma determinada área. Quando o dispositivo recebe o sinal, ele pode acionar um aplicativo para executar uma determinada ação, como um anúncio ou uma oferta em um produto específico, como mostrado na Figura 1.

Com a introdução do iOS 7, a Apple anunciou uma nova tecnologia chamada *iBeacon* (APPLE, INC., 2014). O protocolo *iBeacon* assumiu um papel de liderança na indústria de *beacons*, com vários fabricantes aplicando o protocolo *iBeacon* aos seus *beacons* BLE. A mensagem de transmissão do *iBeacon* é composta de três IDs diferentes, contendo um identificador único universal (UUID - *universally unique identifier*) para identificação do dispositivo ou aplicação e os campos *Major* e *Minor* que podem ser usados para definir sub-regiões com o mesmo UUID, veja a Figura 2 abaixo.

Para garantir uma maior precisão, é importante considerar a força do sinal. Um *iBeacon* fornece uma indicação de intensidade de sinal (RSSI - *Received Signal Strength Indication*), que pode ser capturado por exemplo por um smartphone, para determinar tanto a proximidade do mesmo com o *beacon*, quanto a precisão dessa estimativa de proximidade. A intensidade desse sinal é geralmente correlacionada



Figura 1 – Funcionamento de um *beacon*

Fonte – Kontakt.io

Field	Size	Description
UUID	16 bytes	Application developers should define a UUID specific to their app and deployment use case.
Major	2 bytes	Further specifies a specific iBeacon and use case. For example, this could define a sub-region within a larger region defined by the UUID.
Minor	2 bytes	Allows further subdivision of region or use case, specified by the application developer.

Figura 2 – Os diferentes IDs de um anúncio *iBeacon*

Fonte – Apple, Inc. (2014)

com a distância que um dispositivo está do sinal. Em uma condição ideal (isto é, com uma linha de visão desobstruída entre a antena de um dispositivo e o *beacon*), quanto mais próxima a pessoa estiver, mais preciso será o resultado (APPLE, INC., 2014), como demonstrando nas Figura 3 e Figura 4.

2.5 M-Hub

O middleware M-Hub pode ser definido, de acordo com Talavera et al. (2015), como um serviço de middleware de IoMT geral executado em um dispositivo móvel pessoal, responsável por descobrir e oportunisticamente conectar a vários *smart objects* acessíveis apenas através de tecnologias WPAN de curto alcance. O smartphone executando uma instância do M-Hub, funciona como gateway para *smart objects*, fornecendo acesso à internet para dispositivos que não podem se

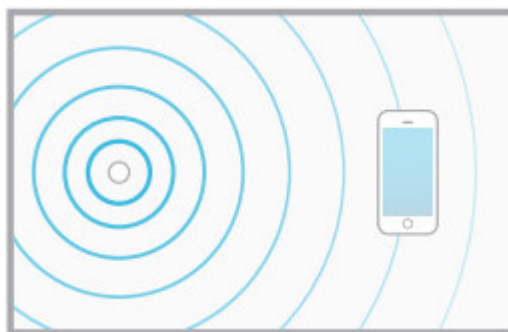


Figura 3 – Quando um dispositivo está mais distante do *beacon*, a intensidade do sinal é diminuída e, portanto, a estimativa de precisão aumentará.

Fonte – Apple, Inc. (2014)

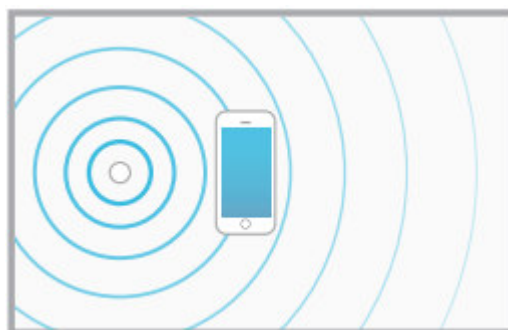


Figura 4 – A intensidade do sinal aumenta à medida que um dispositivo se aproxima do *beacon*, levando a uma melhor estimativa de proximidade.

Fonte – Apple, Inc. (2014)

conectar. Outro recurso importante que pode ser explorado pelas aplicações é a habilidade de enriquecer os dados de sensores com dados de contexto obtidos dos sensores internos do M-Hub.

2.6 Arquitetura Orientada a Serviços

A Arquitetura Orientada a Serviços, do inglês *Service Oriented Architecture* (SOA), é um estilo arquitetural que suporta a orientação a serviços. A orientação a serviço é uma maneira de pensar em termos de serviços, de desenvolvimento baseado em serviços e dos resultados dos serviços. Um serviço é uma representação lógica de uma atividade de negócios repetitiva que tem uma saída específica (por exemplo, checar o crédito do cliente, disponibilizar dados do clima, entre outros) e é independente (o que o serviço precisa para operar deve estar contido nele). Um estilo arquitetural é a combinação de características distintas nas quais a arquitetura é realizada ou representada (GROUP, 2009).

SOA representa um modelo em que a lógica de negócios é decomposta em unidades menores e encoraja essas unidades individuais de lógica a existirem autonomamente mas não isoladas umas das outras. Coletivamente, essas unidades compreendem a lógica de negócios e, individualmente, podem estar distribuídas.

Unidades de lógica precisam seguir um conjunto de princípios que possibilitem que cada unidade evolua de forma independente, mantendo ainda uma quantidade suficiente de uniformização e padronização. Em SOA, essas unidades de lógica são conhecidas como serviços (ERL, 1900).

Para manter a independência, os serviços encapsulam a lógica dentro de um contexto distinto. Esse contexto pode ser específico para uma tarefa de negócios, uma entidade de negócios ou algum outro agrupamento lógico. A responsabilidade atribuída a um serviço pode ser pequena ou grande. Portanto, o tamanho e o escopo da lógica representados pelo serviço podem variar (ERL, 1900).

2.6.1 Serviços Web

Um serviço web é uma parte de uma lógica de negócio, localizado em algum lugar da Internet, que é acessível por protocolos de Internet baseados em padrões como o *HyperText Transfer Protocol* (HTTP). O uso de um serviço web pode ser algo simples como a realização de autenticação em um site ou algo complexo como facilitar a operação de negócios de uma grande empresa (CHAPPELL; JEWELL, 2002).

Serviços web, na perspectiva da Arquitetura Orientada a Serviços (SOA), possuem as seguintes características principais: são encapsulados e fracamente acoplados (RICCI; DENTI; PIUNTI, 2010).

Segundo Ricci, Denti e Piunti (2010), serviços web representam uma tecnologia de referência para a criação de sistemas distribuídos, que por sua vez, precisam suportar interoperabilidade entre aplicações heterogêneas distribuídas por meio de uma rede.

2.6.1.1 Serviços Web RESTful

Representational State Transfer (REST) é um estilo arquitetural para o desenvolvimento de serviços web originado no trabalho de Fielding e Taylor (2000), que também é coautor do protocolo HTTP. Dessa forma, o estilo arquitetural REST é guiado, dentre outros preceitos, pelas boas práticas de uso de HTTP: uso adequado dos métodos e cabeçalhos HTTP, uso adequado de *Uniform Resource Locator* (URL), uso de códigos de status padronizados para representação de sucesso ou falha e interligações entre vários recursos diferentes (SAUDATE, 2014).

REST é baseado em recursos, que são os conjuntos de dados trafegados pelo protocolo. Para representar dados estruturados, em serviços REST, são usados *eXtensible Markup Language* (XML) ou *JavaScript Object Notation* (JSON) na maioria dos casos. Cada recurso é representado por *Uniform Resource Identifier* (URI), um endereço próprio. Na web, URIs e URLs são essencialmente a mesma coisa (SAUDATE, 2014).

Os serviços web RESTful são caracterizados por seguirem o estilo arquitetural REST. Utilizando serviços web RESTful, o desenvolvimento da comunicação entre o cliente e o servidor para uma aplicação será o mesmo, independentemente de dispositivo, baseada no protocolo HTTP.

3 SISTEMA DE GESTÃO DE PRESENÇA

3.1 Contextualização do Sistema

O Sistema de Gestão de Presença baseia-se na utilização de diversos serviços para o seu funcionamento, sendo este capaz de captar, armazenar e fornecer dados pertinentes sobre a presença e encontro entre pessoas em determinados espaços físicos.

3.2 Justificativa

Abaixo estão três cenários que mostram exemplos onde o Sistema de Gestão de Presença pode ser utilizado.

- a) Em um hospital, diversos pacientes são assistidos por diversos enfermeiros, médicos e outros profissionais da saúde. Cada paciente fica alocado em um leito e sala específico, então deseja-se ter o controle de quais profissionais estão em contato com um paciente ou em quais salas um profissional atendeu algum paciente.
- b) Em uma empresa de tecnologia, inúmeros funcionários trabalham diariamente, mas quando os funcionários não estão em suas salas, eles podem estar realizando trabalhos externamente. Um sistema de anúncios pode ser programado para enviar uma mensagem de aviso para todos os funcionários presentes atualmente na sala de suporte ou enviar uma mensagem de falha no serviço para o primeiro funcionário que chegar na sala de sistemas.
- c) Em uma universidade, vários laboratórios são disponibilizados para os estudantes, cada professor é responsável por um laboratório, que pode ter várias salas, e por um grupo de alunos a serem orientados. Um professor deseja saber quais de seus alunos estão em seu laboratório no momento ou quais alunos vieram para uma reunião.

3.3 Objetivos do Sistema

O objetivo do projeto é fornecer uma solução de software capaz de registrar a presença de pessoas cadastradas no sistema, em espaços físicos também cadastrados, através da associação de dispositivos móveis (por exemplo, smartphones e *tablets*) à dispositivos *smart objects* (por exemplo, *beacons*, sensores e atuadores), fornecendo uma interface para consultas filtradas de presença ou encontro de pessoas.

3.4 Arquitetura do Sistema

O Sistema de Gestão de Presença propõe a criação de um conjunto de serviços baseado no estilo arquitetural da Arquitetura Orientada a Serviços (SOA) a fim de apoiar o desenvolvimento de sistemas que necessitem da gestão de presença ou de encontro entre pessoas em espaços físicos. Propõe também o uso de um servidor web para o armazenamento e centralização de informações, de forma que o servidor esteja acessível para os dispositivos por meio da Internet. Esses serviços disponibilizam suas respectivas APIs para comunicação com o servidor web, por exemplo, por meio de serviços web RESTful.

3.4.1 Visão Geral do Componentes

Para este trabalho, alguns componentes de software foram escolhidos como os serviços propostos na Arquitetura do Projeto, incluindo uma aplicação Android que serve como cliente do sistema.

- Uma aplicação Android com um módulo M-Hub/CDDL para cadastrar os encontros e um módulo de consulta de presença.
- Um serviço que armazena e processa encontros de dispositivos móveis com *smart objects*, sendo escolhido o *middleware HORYS*.
- Um serviço semântico, que armazena dados de pessoas, espaços físicos, dispositivos móveis e *smart objects*, retornando a associação desses dados, sendo escolhido o *Servidor Semântico*.
- Como o principal objeto deste trabalho, o serviço proposto foi chamado de *Serviço Atendente*, capaz de prover informações necessárias para a gestão de presenças e encontros de pessoas em espaços físicos através de dados disponibilizados pelos serviços HORYS e Servidor Semântico.

A arquitetura proposta do Sistema de Gestão de Presença está representada no diagrama da Figura 5, utilizando os componentes escolhidos.

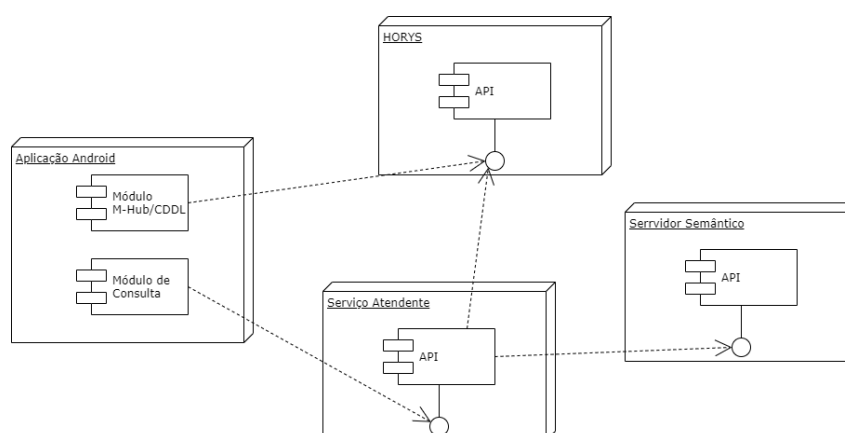


Figura 5 – Arquitetura proposta do projeto.

Fonte – Produzido pelo autor

3.4.2 HORYS

O HORYS (*Hub-Object Rendezvous RegistrY Service*) é um serviço de middleware para armazenamento e processamento de encontros (*rendezvous*) de dispositivos móveis (*mobile hubs*) (por exemplo, smartphones e tablets) com *smart objects* (por exemplo, *beacons*, sensores e atuadores) desenvolvido por Marcos Roriz Jr em parceria com o *Laboratory for Advanced Collaboration* da PUC-Rio.

Além de armazenar o encontro, o HORYS fornece uma API para consultar por exemplo, quais *smart objects* um determinado *mobile hub* encontrou ou quais *mobile hubs* se encontram próximos de um *smart object* em particular. O HORYS utiliza a tecnologia NoSQL MongoDB para armazenar os dados dos *rendezvous* e executar pesquisas otimizadas e paralelas nesse armazenamento de dados (ENDLER; SILVA, 2018).

Os eventos de *rendezvous* ocorrem quando um *mobile hub* (com um dado *UUID hubID*) se encontra no alcance de um determinado *smart object* que também contém um *UUID thingID* único. O HORYS armazena o tempo (em *timestamp*) de tal encontro e também a intensidade do sinal de conexão entre o *mobile hub* e um determinado *smart object*. Em adição, ele também possibilita armazenar a localização (latitude e longitude) onde o evento *rendezvous* aconteceu, como ilustrado na Figura 6. Um exemplo de um JSON representando um *rendezvous* e enviado para o HORYS via POST pode ser encontrado na Figura 7.

mhubID (UUID)	thingID (UUID)	latitude (double)	longitude (double)	signal (double)	timestamp (long)
399eba78- 2346-4c20- 81ba- 0a8cad5fdd22	953e8ccd-c4aa- 4a20-9f28- 4ef90f27a643	-23.01641146	-10.5	-10.5	1494443744

Figura 6 – Estrutura de dados utilizada pelo HORYS.

Fonte – HORYS

```
{
  "appID": "d8be7202-966e-451d-823c-65281b474db1",
  "mhubID": "9fe8edfe-5105-40be-9a4d-78f870d41ea1",
  "thingID": "7d53edbe-2feb-4e63-a1d5-64334587a2df",
  "latitude": 29.2542,
  "longitude": 68.7299,
  "signal": -86.0,
  "timestamp": 1494446451
}
```

Figura 7 – Exemplo de JSON de um *rendezvous* enviado para o HORYS.

Fonte – HORYS

3.4.3 Servidor Semântico

O Servidor Semântico é um serviço web que mapeia os relacionamentos entre dispositivos sem fio com pessoas ou espaços físicos, desenvolvido por Alysso Cirilo Silva no Laboratório de Sistemas Distribuídos Inteligentes (LSDi) da Universidade Federal do Maranhão (UFMA).

Como funcionalidades o Servidor Semântico armazena um conjunto de dados referentes a dispositivos, espaços físicos, pessoas, cargos e associação entre esses, como mostrado no diagrama da Figura 8.

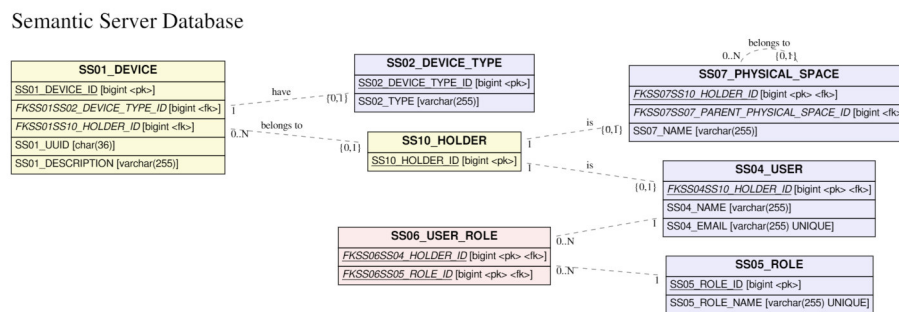


Figura 8 – Diagrama de Entidade Relacionamento do Servidor Semântico.

Fonte – Alysso Cirilo Silva

3.5 Requisitos do Sistema de Gestão de Presença

Como requisitos do projeto, são necessários dispositivos móveis e *smart objects*, aplicações para dispositivos móveis, uma capaz de ler sinais *Bluetooth* de *beacons* e outra capaz de ler e apresentar dados fornecidos pelo Sistema de Gestão de Presença ao usuário final, além de diversos softwares, que atuam como serviços para o mesmo.

Os cenários apresentados na seção 3.2 são a motivação para o desenvolvimento do Sistema de Gestão de Presença. Os requisitos desse foram identificados tomando base estes cenários e as perguntas pertinentes apresentadas, definindo as seguintes buscas:

- Quem são as pessoas presentes em um espaço físico específico no momento atual?
- Quem são as pessoas presentes em um espaço físico específico em um período de tempo específico?
- Dado um grupo de pessoas, quais delas se encontraram, ou seja, estavam presentes no mesmo local e na mesma faixa de tempo?
- Em quais espaços físicos as pessoas específicas estão agora?
- Em quais espaços físicos as pessoas específicas estiveram em um período de tempo específico?

3.5.1 Requisitos Funcionais

A tabela 1 contém uma visão mais formal de todos os requisitos funcionais identificados para o Sistema de Gestão de Presença.

ID	Requisito
1	O sistema deve permitir a inclusão e consulta de um evento <i>rendezvous</i> de dispositivos móveis (<i>mobile hubs</i>) com <i>smart objects</i> .
2	O sistema deve permitir a inclusão e consulta de dados semânticos (dispositivos, espaços físicos, pessoas, cargos, etc).
3	O sistema deve permitir a consulta de informações necessárias para a gestão de presenças e encontros de pessoas em espaços físicos (vide buscas da seção 3.5).

Tabela 1 – Os requisitos funcionais identificados.

Fonte – Produzido pelo autor

3.5.2 Requisitos Não Funcionais

A tabela 2 contém os requisitos não funcionais.

ID	Requisito
1	O sistema deve ser desenvolvido utilizando o serviço HORYS para armazenamento e consulta de dados de encontros (<i>rendezvous</i>).
2	O sistema deve ter APIs para inclusão e consulta de dados.
3	O sistema pode ser desenvolvido utilizando a tecnologia REST como forma de comunicação com as APIs.
4	O sistema deve funcionar em sistemas baseados em GNU/Linux para ser hospedado em um servidor.

Tabela 2 – Os requisitos não funcionais identificados.

Fonte – Produzido pelo autor

4 SERVIÇO ATENDENTE

O serviço desenvolvido, aqui chamado de Serviço Atendente, é um componente de software capaz de prover informações necessárias para a gestão de presenças e encontros de pessoas em espaços físicos através de dados disponibilizados por outros serviços presentes no Sistema de Gestão de Presença.

4.1 Especificação dos Requisitos

As buscas apresentadas na seção 3.5 são a motivação para o desenvolvimento deste serviço. Os requisitos para o Serviço Atendente foram identificados tomando base estas buscas e os requisitos apresentados para o Sistema de Gestão de Presença.

4.1.1 Requisitos Funcionais

A tabela 3 contém uma visão mais formal de todos os requisitos funcionais identificados para o Serviço Atendente.

ID	Requisito
1	O serviço deve ser capaz de se comunicar via API com o HORYS para obtenção de dados sobre encontros (<i>rendezvous</i>).
2	O serviço deve ser capaz de se comunicar via API com o Servidor Semântico para obtenção de dados sobre pessoas, espaços físicos e seus respectivos dispositivos.
3	O serviço deve providenciar uma API para fornecimento de dados gerados a partir dos serviços HORYS e Servidor Semântico.
4	O serviço deve fornecer uma rotina que informe as pessoas presentes em um determinado espaço físico no momento atual.
5	O serviço deve fornecer uma rotina que informe se um grupo de pessoas estiveram presentes em um mesmo espaço físico na mesma faixa de tempo.
6	O serviço deve fornecer uma rotina que informe as pessoas presentes em um determinado espaço físico em uma faixa de tempo.
7	O serviço deve fornecer uma rotina que informe os espaços físicos em que determinada pessoa está presente no momento atual.
8	O serviço deve fornecer uma rotina que informe os espaços físicos em que determinada pessoa estivera presente em uma faixa de tempo.

Tabela 3 – Os requisitos funcionais identificados.

Fonte – Produzido pelo autor

4.1.2 Requisitos Não Funcionais

A tabela 4 contém os requisitos não funcionais.

ID	Requisito
1	O sistema deve ser desenvolvido utilizando o serviço HORYS para armazenamento e consulta de dados de encontros (<i>rendezvous</i>).
2	O sistema deve ter APIs para inclusão e consulta de dados.
3	O sistema pode ser desenvolvido utilizando a tecnologia REST como forma de comunicação com as APIs.
4	O sistema deve funcionar em sistemas baseados em GNU/Linux para ser hospedado em um servidor.
5	O sistema deve processar requisições de forma paralela para poder ser acessado por vários usuários ao mesmo tempo.

Tabela 4 – Os requisitos não funcionais identificados.

Fonte – Produzido pelo autor

4.2 Arquitetura do Serviço

Este trabalho propõe a criação de uma API baseada em Java a fim de apoiar o desenvolvimento de aplicações que necessitem de informações para a gestão de presença e encontro entre pessoas em espaços físicos, baseando-se em dados originados de dois outros serviços, um que armazena dados de encontros entre dispositivos e outro que armazena dados semânticos de pessoas, espaços físicos e seus respectivos dispositivos.

As funcionalidades da API se baseiam nas buscas definidas para o Sistema de Gestão de Presença, encontradas na seção 3.5. A comunicação da API com o servidor web se dá por meio de serviços web RESTful. Desta forma, o cliente (uma aplicação Android, por exemplo) deve utilizar uma requisição GET HTTP sobre uma URL definida pela API do Serviço Atendente para obtenção dos dados requisitados.

4.3 Interface

Para a interface, o Serviço Atendente disponibiliza uma API, utilizando-se da tecnologia REST, com requisições HTTP do tipo GET, definida a seguir.

GET /physical_spaces/{id/...}/persons

Essa operação recupera todas as pessoas presentes agora nos espaços físicos com id presente na lista *{id/...}*.

Exemplo de parâmetros de entrada:

http://localhost:8080/service/physical_spaces/3/persons/

Exemplo de parâmetros de saída:

```
1 [
2   {
3     "shortName": "Daniel Carvalho",
4     "email": "daniel.carvalho@lsdi.ufma.br",
5     "physical_space": "LSDi",
6     "duration": 3600
7   },
8   {
9     "shortName": "Andre Luiz",
10    "email": "andreluizalmeidacardoso@gmail.com",
11    "physical_space": "Sala das ETs",
12    "duration": 7200
13  }
14 ]
```

`GET /physical_spaces/{id/...}/persons/{Q}/{W}`

Essa operação recupera todas as pessoas presentes nos espaços físicos com `id` presente na lista `{id/...}` no intervalo de tempo `{Q}` a `{W}`.

Exemplo de parâmetros de entrada:

`http://localhost:8080/service/physical_spaces/3/persons/1558764156/1559764156/`

Exemplo de parâmetros de saída:

```
1 [
2   {
3     "shortName": "Daniel Carvalho",
4     "email": "daniel.carvalho@lsdi.ufma.br",
5     "physical_space": "LSDi",
6     "arrive": 1559007804,
7     "depart": 1559007837
8   },
9   {
10    "shortName": "Andre Luiz",
11    "email": "andreluizalmeidacardoso@gmail.com",
12    "physical_space": "Sala das ETs",
13    "arrive": 1559332894,
14    "depart": 1559332904
15  }
16 ]
```

`GET /persons/{id/...}/rendezvous/{Q}/{W}`

Essa operação recupera um grupo de pessoas que estavam presentes em um espaço físico no mesmo lapso de tempo com `id` presente na lista `{id/...}` no intervalo de tempo `{Q}` a `{W}`.

Exemplo de parâmetros de entrada:

<http://localhost:8080/service/persons/1/2/rendezvous/1551113150/1551287880/>

Exemplo de parâmetros de saída:

```
1 [
2   {
3     "physical_space": "Sala das ETs",
4     "persons": [
5       {
6         "shortName": "Alyson"
7       },
8       {
9         "shortName": "Daniel"
10      }
11    ],
12    "arrive": 1551113190,
13    "depart": 1551113208
14  }
15 ]
```

GET /persons/{id/...}/physical_spaces

Essa operação recupera todos os espaços físicos onde as pessoas existentes com id presente na lista {id/...} estão presentes.

Exemplo de parâmetros de entrada:

http://localhost:8080/service/persons/1/2/physical_spaces/

Exemplo de parâmetros de saída:

```
1 [
2   {
3     "shortName": "Daniel",
4     "physical_space": "Sala das ETs",
5     "description": "Sala das Estacoes de Trabalho do LSDi",
6     "duration": 3600
7   },
8   {
9     "shortName": "Alyson",
10    "physical_space": "Sala das ETs",
11    "description": "Sala das Estacoes de Trabalho do LSDi",
12    "duration": 7200
13  }
14 ]
```

`GET /persons/{id/...}/physical_spaces/{Q}/{W}`

Essa operação recupera todos os espaços físicos onde as pessoas existentes com id presente na lista `{id/...}` estavam presentes no intervalo de tempo `{Q}` a `{W}`.

Exemplo de parâmetros de entrada:

`http://localhost:8080/service/persons/1/2/physical_spaces/1551113150/1551287880/`

Exemplo de parâmetros de saída:

```
1 [
2   {
3     "shortName": "Daniel",
4     "physical_space": "Sala das ETs",
5     "description": "Sala das Estacoes de Trabalho do LSDi",
6     "arrive": 1551287871,
7     "depart": 1551287876
8   },
9   {
10    "shortName": "Daniel",
11    "physical_space": "Sala das ETs",
12    "description": "Sala das Estacoes de Trabalho do LSDi",
13    "arrive": 1551113153,
14    "depart": 1551113208
15  },
16  {
17    "name": "Alyson",
18    "physical_space": "Sala de Reunioes",
19    "description": "Sala de Reunioes do LSDi",
20    "arrive": 1551113190,
21    "depart": 1551113220
22  }
23 ]
```

4.3.1 Exemplo com Diagrama de Atividade

O diagrama de atividade na Figura 9 mostra o fluxo do serviço em um caso de uso exemplificando um usuário pesquisando as pessoas presentes em um espaço físico, representada na API pela operação `GET /physical_spaces/{id/...}/persons`. Começa quando o usuário realiza uma requisição GET via HTTP, utilizando a chamada correspondente da API e passando um espaço físico específico. O Serviço Atendente identifica a requisição e realiza uma busca no Servidor Semântico pelo espaço físico. Retornado o espaço físico, uma nova busca é feita no Servidor Semântico, dessa vez pelos *beacons* presentes no mesmo. Em posse de uma lista de *beacons*, o serviço realiza uma chamada ao HORYS, pedindo as durações dos encontros em que os *beacons* estiveram. Se estiverem havendo *rendezvous* com estes *beacons*, é retornada uma lista com *mhubs*

e durações. Com essa lista de *mhubs*, o serviço realiza buscas no Servidor Semântico para saber se esses *mhubs* existem, e que pessoas possuem eles. Por fim, o Serviço Atendente retorna uma lista contendo pessoa, espaço físico e duração da presença.

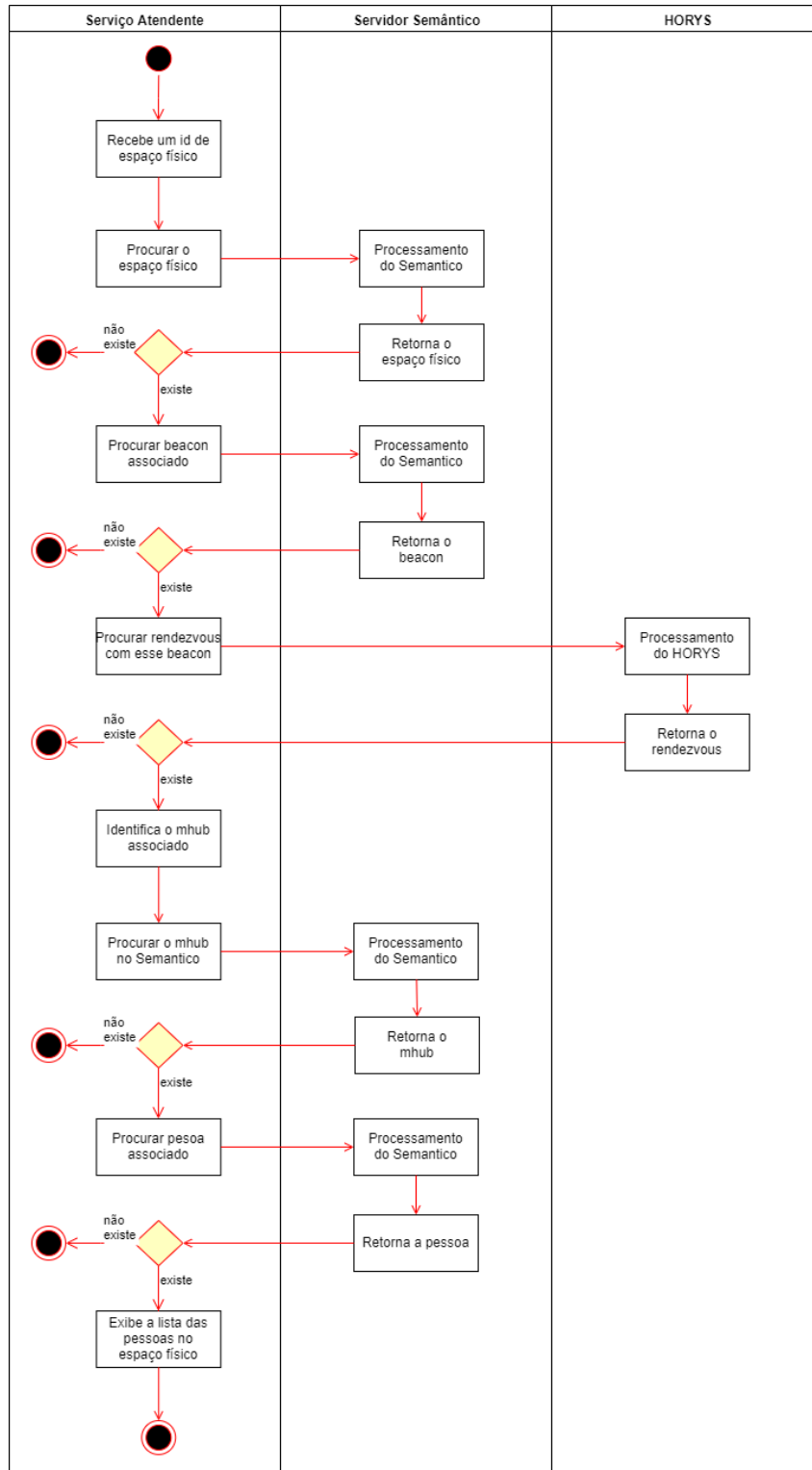


Figura 9 – Fluxograma de um caso de uso específico.

Fonte – Produzido pelo autor

4.4 Aspectos de Implementação

Essa seção trata sobre as ferramentas e tecnologias utilizadas, bem como aspectos da implementação do serviço proposto.

4.4.1 Ferramentas e Tecnologias

Para a criação do serviço proposto, foi definida a utilização de um *Web Service RESTful*, ou seja, um serviço web que utiliza a arquitetura REST. Para isso foram utilizadas as seguintes ferramentas e tecnologias:

- a) Java:
 - uma linguagem de programação de uso geral baseada em classes, orientada a objetos e projetada para ter o menor número possível de dependências de implementação.
- b) JAX-RS (*Java API for RESTful Web Services*):
 - uma biblioteca da linguagem de programação Java que fornece suporte à criação de serviços web com o padrão arquitetural REST.
- c) Glassfish:
 - um servidor de aplicação *open source* para a plataforma Java.

4.4.2 Implementação

O Serviço Atendente foi desenvolvido como um serviço web *RESTful* utilizando a ferramenta JAX-RS da linguagem de programação JAVA. O código-fonte foi subdividido em pacotes, com o pacote *core* contendo classes principais de configuração da aplicação e definição dos recursos web, o pacote *database* contendo classes de comunicação com um banco de dados ou com outros serviços, e o pacote *data* contendo classes dos tipos de dados utilizados pelo serviço. O diagrama da Figura 10 ilustra a organização dos pacotes no código-fonte.

No diagrama da Figura 11 podemos identificar a classe *ApplicationConfig*, uma extensão da classe *Application* da biblioteca JAX-RS vista na subseção 4.4.1 através do mecanismo de herança, que adiciona à aplicação um recurso REST representado pela classe *ServiceResource*, classe essa responsável pelos recursos REST do serviço. O Serviço Atendente possui 2 modos de operação, no modo 0 padrão os espaços físicos possuem *beacons* e as pessoas possuem *mobile hubs*, no modo 1 os espaços físicos possuem *mobile hubs* e as pessoas possuem *beacons*.

No diagrama da Figura 12 podemos observar as classes de acesso às base de dados. A classe *RendezvousDaoImpHorys* implementa a interface *RendezvousDAO* e possui os métodos utilizados para aquisição de dados de encontros disponibilizados pelo HORYS. A classe *ServiceDaoImpSemantic* implementa a interface *ServiceDao* e possui os métodos utilizados para aquisição de dados de mapeamento de pessoas e espaços físicos para dispositivos disponibilizados pelo Servidor Semântico. A classe *REST* implementa a operação HTTP GET utilizada por outras classes.

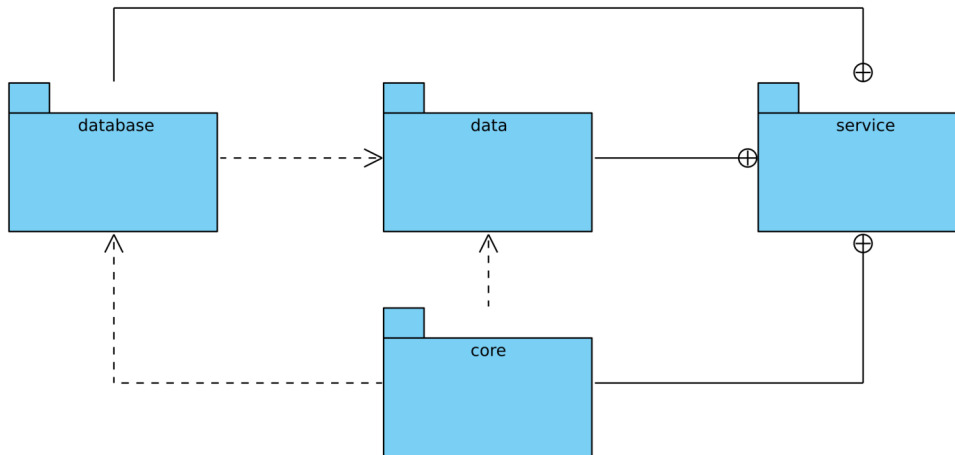


Figura 10 – Diagrama de pacotes.

Fonte – Produzido pelo autor

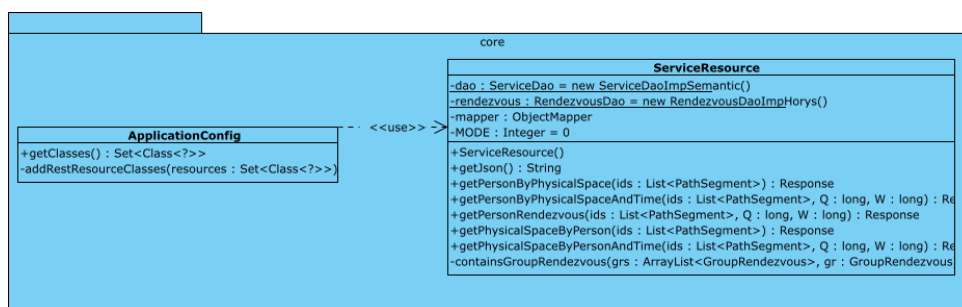


Figura 11 – Diagrama de classes do pacote core.

Fonte – Produzido pelo autor

No diagrama da Figura 13 podemos encontrar as classes de dados utilizadas pelo Serviço Atendente sendo que *Person* representa uma pessoa, *Physical Space* representa um espaço físico, *Device* representa um dispositivo que pode ser um *mhub* ou um *thing*, *Rendezvous* representa um encontro e *GroupRendezvous* representa um encontro entre grupo de pessoas.

A classe *ServiceResource* implementa a API disponibilizada pelo Serviço Atendente, mostrada na seção 4.3.

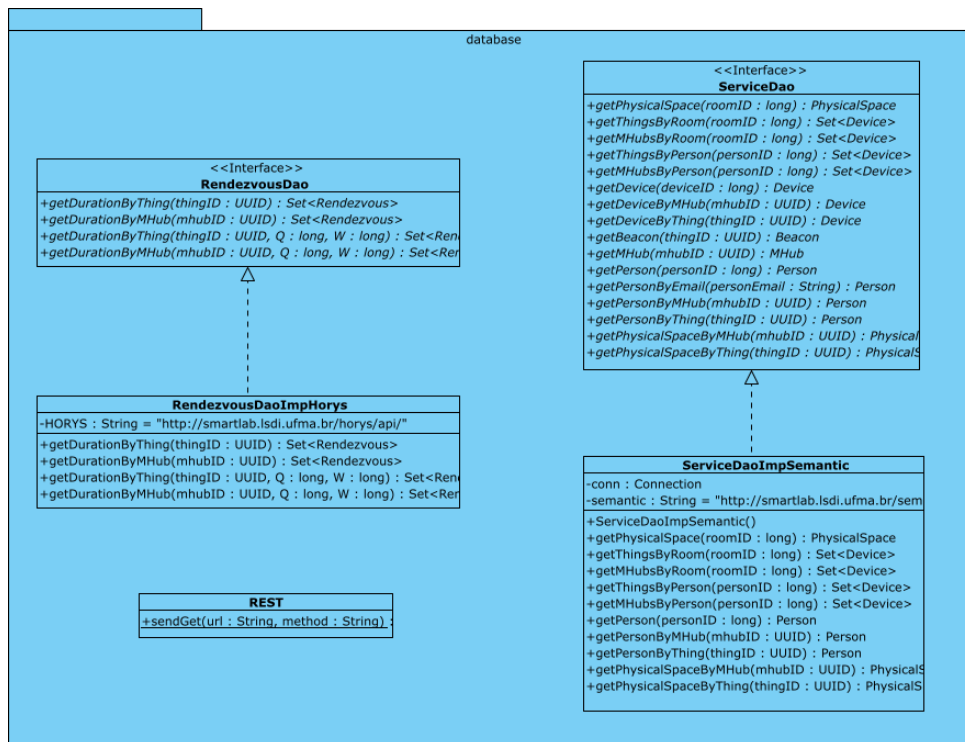


Figura 12 – Diagrama de classes do pacote *database*.

Fonte – Produzido pelo autor

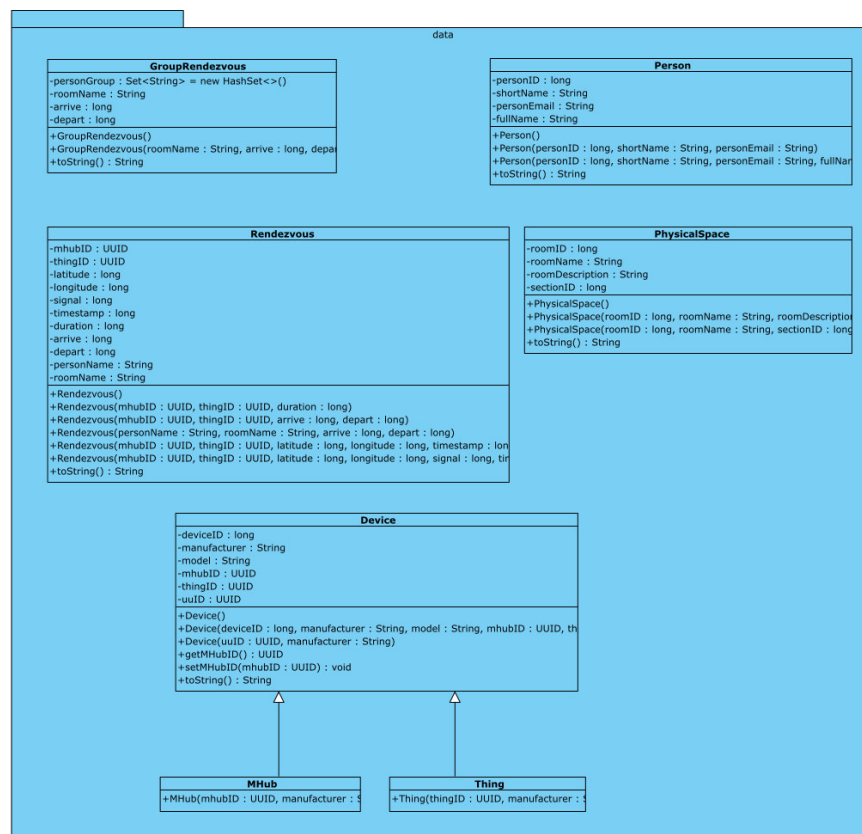


Figura 13 – Diagrama de classes do pacote *data*.

Fonte – Produzido pelo autor

5 AVALIAÇÃO

O objetivo deste capítulo é descrever as avaliações utilizadas na solução proposta. A avaliação tem como objetivo determinar a performance da implementação do serviço proposto, chamado aqui de Serviço Atendente, rodando em conjunto com serviços fornecedores de dados de encontros e dados de mapeamento semântico. Em todos os experimentos, utilizou-se o HORYS e o Servidor Semântico.

A avaliação foi realizada por meio de experimentos, onde a performance será avaliada através da contagem de tempo desde que uma requisição é feita ao Serviço Atendente até o momento em que o mesmo retorna um resultado.

5.1 Experimento 1

5.1.1 Métricas

A fim de determinar a performance da solução proposta, a aplicação permanece constantemente anotando os *timestamps* no começo e fim do processamento, ambos em milissegundos. O primeiro *timestamp* é subtraído do segundo de forma a calcular o tempo de retorno total do sistema composto dos serviços Serviço Atendente, HORYS e Servidor Semântico, que será de agora em diante referido como Δt_{total} e calculado conforme a Equação 5.1.

$$\Delta t_{total} = t_{retorno_requisicao} - t_{inicio_requisicao} \quad (5.1)$$

Onde $t_{retorno_requisicao}$ e $t_{inicio_requisicao}$ se referem ao *timestamp* em que o processamento foi finalizado e a requisição retornará um resultado e ao *timestamp* em que a requisição é feita e será então feito o processamento, respectivamente.

5.1.2 Cenário

Para medir os parâmetros anteriores, foi desenvolvido um cenário de uso, onde será possível determinar tais aspectos.

Este cenário de uso consiste em uma casa onde cada cômodo é equipado com um *beacon* BLE, calibrado de forma que o sinal emitido não possa ser detectado fora do cômodo e configurado para emitir 10 anúncios por segundo. Uma pessoa portando um smartphone é instruída a conduzir as atividades do dia a dia nesta residência. Este smartphone executa uma aplicação desenvolvida com o middleware M-Hub/CDDL que detecta os anúncios dos *beacons*, determinando em qual região da casa o indivíduo se encontra.

Ao entrar em um cômodo, o *beacon* será encontrado pelo M-Hub/CDDL, gerando um evento de encontro (*rendezvous*), que deverá ser enviado ao HORYS.

5.1.3 Simulação de Locomoção na Residência

O cenário da locomoção de um indivíduo em uma residência como descrito na subseção 5.1.2 foi simulado utilizando um *dataset*. Os dados utilizados para a simulação deste experimento foram obtidos do *dataset* disponibilizado por Byrne et al. (2018). Neste trabalho os autores instruíram que participantes conduzissem suas rotinas diárias em casa, enquanto sua localização na residência era monitorada.

O chão dos cômodos das casas foi marcado com etiquetas, onde cada etiqueta é uma imagem binária que codifica um número inteiro, e este, único para cada uma das etiquetas. Os participantes são equipados com uma câmera na região do torso que aponta em direção ao chão, detectando e interpretando qual etiqueta está visível no momento, e deste modo, identificando em qual cômodo o participante se encontra.

No trabalho citado, o autor realizou o experimento em 4 residências distintas, e estas foram denotadas de “Residence A”, “Residence B”, “Residence C” e “Residence D”. Para cada residência, o experimento foi conduzido múltiplas vezes, e cada um denominado de “living_1”, “living_2”, “living_3”, ..., “living_n”.

A fim de maximizar a quantidade de eventos de descoberta gerados na simulação, foi utilizado os dados do experimento “living_2” da casa “Residence D” pois este experimento possui um dos maiores tempos de monitoramento e quantidade de entrada em cômodos em todo o *dataset*. Os dados deste experimento consistem de 58 minutos de monitoramento de um indivíduo em uma casa de 10 cômodos. Os dados possuem uma resolução média de aproximadamente 11 medições por segundo.

O *dataset* original é composto principalmente por um arquivo de texto que contém em cada linha uma representação de um momento no decorrer do experimento. Entre os dados contidos por linha, destaca-se o *timestamp* da medição e a etiqueta detectada naquele instante.

Realizou-se um pré-processamento no *dataset* de forma a facilitar a utilização dos dados para fim da simulação, gerando um arquivo contendo 3 valores separadas por espaço em que cada linha representa o instante em que a medição foi feita, representando um *rendezvous*. Os 3 valores são: o tempo em que a medição foi feita, um identificador UUID do *beacon* presente no cômodo e um identificador UUID do dispositivo móvel (*mhub*) de um residente, como pode ser visto no Código 1.

Código 1 – Parte do *dataset* pré-processado

	#time	thingID	mhubID
1			
2	14:47:01	5f0125a0-2efa-40fe-a73e-0a0a1feb2637	aa9b1a81-6de6-4e85-824b-924e29cb2bff
3	14:47:01	5f0125a0-2efa-40fe-a73e-0a0a1feb2637	aa9b1a81-6de6-4e85-824b-924e29cb2bff
4	14:47:01	5f0125a0-2efa-40fe-a73e-0a0a1feb2637	aa9b1a81-6de6-4e85-824b-924e29cb2bff
5	14:47:01	5f0125a0-2efa-40fe-a73e-0a0a1feb2637	aa9b1a81-6de6-4e85-824b-924e29cb2bff
6	14:47:02	5f0125a0-2efa-40fe-a73e-0a0a1feb2637	aa9b1a81-6de6-4e85-824b-924e29cb2bff
7	14:47:02	5f0125a0-2efa-40fe-a73e-0a0a1feb2637	aa9b1a81-6de6-4e85-824b-924e29cb2bff
8	14:47:02	5f0125a0-2efa-40fe-a73e-0a0a1feb2637	aa9b1a81-6de6-4e85-824b-924e29cb2bff
9	14:47:02	5f0125a0-2efa-40fe-a73e-0a0a1feb2637	aa9b1a81-6de6-4e85-824b-924e29cb2bff
10	14:47:03	eae6f624-3e3a-403c-855e-27650053570d	aa9b1a81-6de6-4e85-824b-924e29cb2bff

5.1.4 Simulação dos *Beacons*

Como descrito anteriormente, o cenário consiste em um *beacon* em cada cômodo de uma casa. Utilizando os dados descritos na subseção 5.1.3 foi realizado uma simulação de uma pessoa portando um smartphone com uma aplicação M-Hub/CDDL que detecta a presença de *beacons* em cada cômodo onde o indivíduo adentra.

Para este experimento, cada linha do dataset pré-processado é utilizada como entrada para um JSON enviado ao HORYS via HTTP POST, como o apresentado na subseção 3.4.2, por um software criado para os experimentos, no horário apresentado pela primeira coluna, ou seja, começando às 14:47:01.

5.1.5 Simulação do Sistema de Gestão de Presença

Para a simulação do sistema, os três serviços HORYS, Servidor Semântico e Serviço Atendente foram executados em *containers* criados utilizando o software Docker com as configurações padrões.

Para o Experimento 1, foi escolhida a requisição *GET /physical_spaces/{id/...}/persons*, feita em três horários escolhidos ao acaso durante o período de simulação dos *beacons*: 14h50min, 15h10min e 15h30min, e anotado os tempos de início e retorno da requisição para o cálculo do Δt_{total} . Essa medição foi feita dez vezes seguidas e seus tempos médios calculados.

5.1.6 Recursos computacionais

O experimento descrito foi executado em um computador rodando o *Windows* 10, com processador AMD Ryzen 5 3600 de 6 núcleos e 3.6GHz, e 16GB de memória RAM.

5.1.7 Resultados

Para apresentação dos resultados e melhor visualização do tempo de processamento de cada serviço, o tempo de retorno Δt_{total} foi subdividido em três tempos em milissegundos, um para o Serviço Atendente, um para o HORYS e um para o Servidor Semântico, apresentados nas tabelas que se seguem:

Tabela 5 – Experimento 1 realizado às 14h50min

Média	Total	Serviço	HORYS	Semântico
Δt_{total} (ms)	56.4	0.9	5.4	50.1

Fonte – Produzido pelo autor

Tabela 6 – Experimento 1 realizado às 15h10min

Média	Total	Serviço	HORYS	Semântico
Δt_{total} (ms)	37.9	0.5	3.6	33.8

Fonte – Produzido pelo autor

Tabela 7 – Experimento 1 realizado às 15h30min

Média	Total	Serviço	HORYS	Semântico
Δt_{total} (ms)	35.2	0.3	3.9	31.0

Fonte – Produzido pelo autor

5.1.8 Análise dos Resultados

Analisando as tabelas 6 e 7, podemos observar que o tempo de resposta Δt_{total} do Serviço Atendente, para o experimento realizado, é muito pequeno para o valor do tempo de resposta de todos os três serviços somados. Já o tempo Δt_{total} do Servidor Semântico é o tempo mais custoso dos três serviços presentes no sistema.

Também podemos identificar que, dependendo do horário em que uma mesma requisição é feita, a quantidade de *rendezvous* capturados pelo HORYS naquele momento é diferente, pois depende de quantas pessoas ou espaços físicos estão interagindo entre si, conseqüentemente o tempo de resposta também muda.

Do ponto de vista de um experimento realizado com poucas pessoas utilizando o sistema ao mesmo tempo, a solução apresentada se mostra satisfatória, necessitando posteriormente de testes em escalas maiores.

5.2 Experimento 2

O segundo experimento visa avaliar a solução proposta utilizando uma outra requisição, dessa vez buscando encontros que já ocorreram em uma determinada faixa de tempo, por isso, mantém-se as mesmas métricas, cenário, simulação de locomoção na residência e simulação dos *beacons*.

5.2.1 Simulação do Sistema de Gestão de Presença

Para a simulação do sistema, os três serviços HORYS, Servidor Semântico e Serviço Atendente foram executados em *containers* criados utilizando o software *Docker* com as configurações padrões.

Para o Experimento 2, foi escolhida a requisição `GET /persons/{id/...}/physical_spaces/{Q}/{W}`, feita após o fim do período de simulação dos *beacons*, passando como parâmetro os *timestamps* de início e fim dessa simulação, e anotado os tempos de início e retorno da requisição para o

cálculo do Δt_{total} . Essa medição foi feita dez vezes seguidas e seus tempos médios calculados.

5.2.2 Resultados

Para apresentação dos resultados e melhor visualização do tempo de processamento de cada serviço, o tempo de retorno Δt_{total} foi subdividido em três tempos em milissegundos, um para o Serviço Atendente, um para o HORYS e um para o Servidor Semântico, apresentados nas tabelas que se seguem:

Tabela 8 – Experimento 2

Média	Total	Serviço	HORYS	Semântico
Δt_{total} (ms)	356.4	4.4	4.6	347.4

Fonte – Produzido pelo autor

5.2.3 Análise dos Resultados

Analisando a tabela 8, podemos observar, assim como no experimento 1, que o tempo de resposta Δt_{total} do Serviço Atendente, é muito pequeno para o valor do tempo de resposta de todos os três serviços somados, já o tempo Δt_{total} do Servidor Semântico é o que mais influencia no resultado final.

5.3 Análise Final

Analisando os dados apresentados em ambos os experimentos, podemos identificar que, para os testes em menor escala, o HORYS apresenta um tempo pequeno, entretanto, em uma situação real com muitos registros de encontros salvos no HORYS, é esperado que o tempo de retorno dele cresça exponencialmente.

Em relação ao tempo de retorno apresentado pelo Servidor Semântico, esse apresentou um tempo de retorno muito maior, analisando o algoritmo, pode-se supor que a principal causa seja o fato de que, embora o HORYS tenha um volume de dados maior, ele é acessado poucas vezes, enquanto o Servidor Semântico é acessado diversas vezes: para cada dispositivo identificado no retorno do HORYS, quatro acessos ao Servidor Semântico são feitos, como mostrado na Figura 9. Uma solução para isso é uma futura modificação no Serviço Atendente de criar um cache de dados semânticos, para que não sejam feitos novos acessos na busca de dados repetidos.

6 CONCLUSÃO

Este trabalho apresentou uma proposta de um serviço para prover informações necessárias para a gestão de presenças e encontros de pessoas em espaços físicos através de dados coletados a partir de *beacons* BLE. No contexto deste trabalho, a gestão de presenças e encontros de pessoas refere-se a identificação de informações relativas à presença física de indivíduos em espaços físicos (como um laboratório, uma sala de aula, ou o consultório de um médico) bem como a co-localização de indivíduos nestes espaços físicos visando ou não a realização de tarefas coletivas.

Como metodologia proposta utilizou-se dados fornecidos por dois softwares externos, o HORYS que guarda e provê informações sobre encontros entre dispositivos e o Servidor Semântico que mapeia dispositivos para pessoas ou espaços físicos, para criar um terceiro software, um serviço que fizesse associação dos dados destes últimos e provesse as informações objetivadas. Esses três software atuando em conjunto como em uma arquitetura orientada a serviços, juntamente com *beacons* e aplicações Android, possibilitam a criação de um sistema geral capaz de gerenciar a presença e encontros de pessoas em espaços físicos, independente do contexto em que esse sistema seja utilizado.

A partir da avaliação dos experimentos é possível perceber que a proposta inicial foi atingida, obtendo-se os resultados das buscas da API de forma eficaz e com mínimo impacto causado pelo serviço proposto no tempo de retorno do sistema para o usuário final. Com essas observações podemos dizer que a metodologia utilizada foi validada e é funcional para a obtenção de informações de localização de pessoas em espaços físicos.

7 Trabalhos Futuros

A partir da metodologia aplicada é possível perceber que esta é uma metodologia promissora, podendo servir de base para o desenvolvimento de novas abordagens semelhantes. Através da ideia principal deste trabalho, pode-se acrescentar ao projeto proposto novas funcionalidades, na forma de novas requisições, para a obtenção de informações novas ou refinadas, como por exemplo os ambientes mais visitados, o perfil de pessoas que visitam determinados ambientes, etc.

Além disso, pode-se acrescentar mais informações às buscas propostas, por exemplo incrementando o serviço prestador de dados de encontros entre dispositivos, como o HORYS, para informar o sinal médio dos encontros entre esses dispositivos, a fim de determinar o quão perto estão um do outro. Podemos também implementar serviços utilizando a metodologia do Processamento de Eventos Complexos (*Complex Event Processing Systems*, ou CEP), com o objetivo de fornecer informações sobre a detecção de chegada ou saída de uma pessoa de um ambiente.

Acrescentando aos experimentos realizados para este trabalho, também é possível realizar experimentos em escalas maiores, utilizando vários ambientes acessados por várias pessoas ao mesmo tempo, de forma a verificar e possivelmente melhorar o tempo de resposta do Sistema de Gestão de Presença.

Referências

- ANTON, A.; RUNBERT, J. *Cross-platform Mobile Development and Internet of Things: Developing a cross-platform mobile application using web technologies to interact with smart things*. 2015. Citado na página 17.
- APPLE, INC. *Getting Started with iBeacon*. 1. ed. [S.l.], 2014. Disponível em: <https://developer.apple.com/ibeacon/Getting-Started-with-iBeacon.pdf>. Citado 3 vezes nas páginas 17, 18 e 19.
- ARFWEDSON, H.; SNEDDON, R. Ericsson's bluetooth modules. *Ericsson Review*, Citeseer, v. 76, n. 4, p. 198–205, 1999. Citado na página 17.
- ATZORI, L.; IERA, A.; MORABITO, G. The internet of things: A survey. *Computer Networks*, Elsevier BV, v. 54, n. 15, p. 2787–2805, out. 2010. Disponível em: <<https://doi.org/10.1016/j.comnet.2010.05.010>>. Citado na página 15.
- BYRNE, D.; KOZLOWSKI, M.; SANTOS-RODRIGUEZ, R.; PIECHOCKI, R.; CRADDOCK, I. Residential wearable rssi and accelerometer measurements with detailed location annotations. *Scientific data*, Nature Publishing Group, v. 5, p. 180168, 2018. Citado na página 36.
- CHAPPELL, D. A.; JEWELL, T. *Java web services*. [S.l.]: Tecniche Nuove, 2002. Citado na página 20.
- ENDLER, M.; SILVA, F. S. E. Past, present and future of the contextnet iomt middleware. *OJIOT*, v. 4, n. 1, p. 7–23, 2018. Citado na página 23.
- ERL, T. *Service-oriented architecture: concepts, technology, and design*. [S.l.]: Pearson Education India, 1900. Citado na página 20.
- FIELDING, R. T.; TAYLOR, R. N. *Architectural styles and the design of network-based software architectures*. [S.l.]: University of California, Irvine Doctoral dissertation, 2000. v. 7. Citado na página 20.
- FORTINO, G.; GUERRIERI, A.; RUSSO, W.; SAVAGLIO, C. Middlewares for smart objects and smart environments: Overview and comparison. In: *Internet of Things*. Springer International Publishing, 2014. p. 1–27. Disponível em: <https://doi.org/10.1007/978-3-319-00491-4_1>. Citado na página 13.
- GROUP, T. O. *Soa source book*. [S.l.]: Van Haren Publishing, 2009. Citado na página 19.
- MATTERN, F.; FLOERKEMEIER, C. From the internet of computers to the internet of things. In: _____. *From Active Data Management to Event-Based Systems and More: Papers in Honor of Alejandro Buchmann on the Occasion of His 60th Birthday*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010. p. 242–259. ISBN 978-3-642-17226-7. Disponível em: <https://doi.org/10.1007/978-3-642-17226-7_15>. Citado na página 13.

NAHRSTEDT, K.; LI, H.; NGUYEN, P.; CHANG, S.; VU, L. Internet of mobile things: Mobility-driven challenges, designs and implementations. In: IEEE. *2016 IEEE First International Conference on Internet-of-Things Design and Implementation (IoTDI)*. [S.l.], 2016. p. 25–36. Citado 2 vezes nas páginas 13 e 16.

RICCI, A.; DENTI, E.; PIUNTI, M. A platform for developing soa/ws applications as open and heterogeneous multi-agent systems. *Multiagent and Grid Systems*, IOS Press, v. 6, n. 2, p. 105–132, 2010. Citado na página 20.

SANTOS, B. P.; SILVA, L.; CELES, C.; BORGES, J. B.; NETO, B. S. P.; VIEIRA, M. A. M.; VIEIRA, L. F. M.; GOUSSEVSKAIA, O. N.; LOUREIRO, A. Internet das coisas: da teoriaa prática. *Minicursos SBRC-Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuidos*, 2016. Citado 3 vezes nas páginas 13, 15 e 17.

SAUDATE, A. *REST: Construa API's inteligentes de maneira simples*. [S.l.]: Editora Casa do Código, 2014. Citado na página 20.

TALAVERA, L. E.; ENDLER, M.; VASCONCELOS, I.; VASCONCELOS, R.; CUNHA, M.; SILVA, F. J. d. S. e. The mobile hub concept: Enabling applications for the internet of mobile things. In: IEEE. *Pervasive computing and communication workshops (PerCom workshops), 2015 IEEE international conference on*. [S.l.], 2015. p. 123–128. Citado na página 18.