UNIVERSIDADE FEDERAL DO MARANHÃO

Programa de Pós-Graduação em Ciência da Computação

Tércio Santana Silva Sousa

Qualidade de Contexto em uma Infraestrutura para Navegação Indoor

São Luís
2019

UNIVERSIDADE FEDERAL DO MARANHÃO
CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Tércio Santana Silva Sousa

Qualidade de Contexto em uma Infraestrutura para Navegação Indoor

São Luís
2019

Tércio Santana Silva Sousa

Qualidade de Contexto em uma Infraestrutura para Navegação Indoor

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal do Maranhão como requisito parcial para a obtenção do grau de MESTRE em Ciência da Computação.

Orientador: Samyr Béliche Vale

Doutor em Ciência da Computação – UFMA

São Luís

2019

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

Sousa, Tércio.
Qualidade de Contexto em uma Infraestrutura para Navegação Indoor / Tércio Sousa. – 2019.
90 p.

Orientador(a): Samyr Béliche Vale.
Dissertação (Mestrado) – Programa de Pós-graduação em Ciência da Computação/ccet, Universidade Federal do Maranhão, São Luís, 2019.

1. Computação Sensível ao Contexto. 2. Computação Ubíqua. 3. Localização Indoor. 4. Qualidade de Contexto. 5. Serviços baseados em Localização. I. Béliche Vale, Samyr. II. Título.

Tércio Santana Silva Sousa

Qualidade de Contexto em uma Infraestrutura para Navegação Indoor

Este exemplar corresponde à redação final da dissertação devidamente corrigida e defendida por Tércio Santana Silva Sousa e aprovada pela comissão examinadora.

Aprovada em 24 de Abril de 2019

BANCA EXAMINADORA

Samyr Béliche Vale (orientador)

Doutor em Ciência da Computação – UFMA

Francisco José da Silva e Silva

Doutor em Ciência da Computação – UFMA

André Castelo Branco Soares

Doutor em Ciência da Computação – UFPI

A Deus por sempre iluminar o meu caminho. À minha família, amigos e professores por todo o apoio oferecido.

Resumo

A computação ubíqua é uma subárea da computação que define ambientes dotados de dispositivos capazes de tornar a interação entre humanos e computadores onipresente e invisível. Já a ciência de contexto descreve aplicações capazes de identificar características do ambiente em que estão inseridas e adaptar o seu comportamento para oferecer serviços personalizados de acordo com estas informações. A união da computação ubíqua com a ciência de contexto possibilitou o desenvolvimento de novas aplicações e áreas de pesquisa como a localização indoor, que tem ganhado atenção nos últimos anos com o intuito de resolver problemas de localização em ambientes internos, visto que o GPS oferece precisão aceitável apenas em ambientes outdoor. Este trabalho foi realizado no contexto da arquitetura INavigS, que fornece serviços para a navegação em ambientes fechados através de dispositivos móveis, como os smartphones. Apesar de oferecer os serviços propostos, foram identificados problemas como a baixa precisão de localização, ausência de dados de contexto que definam a orientação do usuário e tempo de resposta alto para exibição de instruções de navegação. O presente trabalho visa propor soluções no âmbito de qualidade de contexto visando melhorar a sua experiência de uso e fornecimento de serviços mais eficientes.

Palavras-chave: Computação Ubíqua, Computação Sensível ao Contexto, Localização Indoor, Qualidade de Contexto, Serviços baseados em Localização, Navegação *indoor*.

Abstract

The ubiquitous computing is a research area of the computer science that defines the environments equipped with devices capable of making the interaction between humans and computers invisible and omnipresent. Context awareness describes the applications able to identify characteristics of the environment in which they are inserted and to adapt their behavior to provide customized services according to this information. The combination of the ubiquitous computing with the context awareness has allowed the development of new applications and research areas such as indoor location, which has gained attention in recent years with the purpose to solve the indoor location problems due to GPS provides accuracy acceptable only in outdoor environments. This study was conducted in the context of the INavigS architecture, which provides services for indoor navigation using mobile devices, such as smartphones. Despite offering the proposed services, problems such as low localization precision, lack of context data that define user orientation and high response time for display of navigational instructions were identified. The present work aims to propose solutions in the context of context quality aiming to improve its experience of use and provision of more efficient services.

Keywords: Ubiquitous Computing, Context-aware Computing, Indoor Localization, Quality of Context, Location based services, Indoor navigation.

Agradecimentos

Primeiramente a Deus, por sempre iluminar o meu caminho e me tornar capaz de chegar até aqui.

Ao meu orientador, Prof. Samyr Vale pela confiança, paciência, disponibilidade e por sempre me motivar nos momentos mais difíceis.

Aos meus pais e aos meus irmãos por, apesar da distância, sempre me apoiarem durante toda esta trajetória.

Aos meus amigos e membros do LSDi-UFMA, os quais compartilhei momentos de alegrias, tristezas e muito aprendizado, ressaltando a importância de Adeilson Marques, Anderson Soares, Allinger Medeiros, Alysson Cirilo, Danne Makleyston, Eduardo Devidson, Fernando Benedito, Ivan Rodrigues, Leonardo Machado e Rodolfo Alves.

Aos meus colegas da área de Informática do IFMA - Campus São Raimundo das Mangabeiras: Bruno Vicente, Sebastião Rodrigues, Sigfran Santana e Jairo Ferraz por possibilitarem o meu afastamento para cursar o mestrado.

À UFMA, ao PPGCC e ao LSDi pela estrutura dada à execução deste trabalho.

"Deus não escolhe os capacitados, capacita os escolhidos. Fazer ou não fazer algo só depende de nossa vontade e perseverança."

Albert Einstein

Lista de Figuras

Lista de Tabelas

Lista de Siglas

API	*Application Programing Interface*.
BLE	*Bluetooth Low Energy*.
DDS	*Data Distribution Service*.
DR	*Dead Reckoning*.
GPS	*Global Positioning System*.
GSM	*Global System for Mobile Communications*.
IMU	*Inertial Measurement Unit*.
INavigS	*Indoor Navigation System*.
KNN	*K-Nearest Neighborg*.
LAC	*Laboratory For Advanced Collaboration*.
LBS	*Location Based Services*.
LEA	*Location Estimation Algorithm*.
LSDi	Laboratório de Sistemas Distribuídos Inteligentes.
LST	*Least Squares Trilateration*.
M-Hub	*Mobile Hub*.
M-OBJ	*Mobile Objects*.
MQTT	*Message Queue Telemetry Transport*.
MR-UDP	*Mobile Realiable UDP*.
PDR	*Pedestrian Dead Reckoning*.

POI *Point of Interest*.

PUC-Rio Pontifícia Universidade Católica do Rio de Janeiro.

QoC *Quality of Context*.

RSSI *Received Signal Strength Indicator*.

SDDL *Scalable Data Distribution Layer*.

SNR *Signal to Noise Ratio*.

ToA *Time of Arrival*.

UFMA Universidade Federal do Maranhão.

UUID *Universally Unique Identifier*.

WKNN *Weighted K-Nearest Neighbor*.

WPAN *Wired Personal Area Network*.

Sumário

1 Introdução

A Computação Ubíqua é uma subárea da Ciência da Computação que define a ideia de ambientes dotados de dispositivos capazes de tornar a interação entre humanos e computadores onipresente e invisível. Para isto, aplicações ubíquas buscam oferecer outras formas de interação que estejam mais próximas das ações naturais das pessoas, como voz, toque ou gestos. Apesar de parecer representar um fenômeno recente, o termo foi utilizado pela primeira vez em 1988 e publicado em 1991 por Weiser [1].

O autor enxergava um futuro em que os lugares frequentados pelas pessoas seriam compostos de pequenos dispositivos utilizados no dia-a-dia, dotados de capacidades computacionais, como a comunicação com outros dispositivos e processamento de dados. Estes dispositivos podem ser sensores, que capturam informações sobre o ambiente, atuadores que recebem comandos e executam ações ou até mesmo objetos triviais como: lâmpadas, termostatos, fechaduras ou qualquer outro que possa funcionar de maneira autônoma ou ser controlado remotamente via tecnologias de redes.

Na época em que foi idealizada, a Computação Ubíqua parecia uma realidade que somente se concretizaria em um futuro muito distante, mas devido a alguns fatores como a evolução das tecnologias de comunicação sem fio e o advento da computação móvel, é possível perceber frequentemente demonstrações práticas de aplicações desta natureza nos dias atuais.

A Computação Sensível ao Contexto é uma área de pesquisa da Computação Ubíqua que estuda a capacidade de aplicações identificarem o contexto em que estão inseridas com o objetivo de adaptar seu comportamento, seja otimizando seu funcionamento ou customizando os serviços oferecidos. O contexto é entendido como qualquer informação que define o estado de uma entidade que interaja com a aplicação, seja ela uma pessoa, um lugar ou um objeto [2].

Os dados que representam o contexto não são totalmente confiáveis. Eles podem ser: imprecisos, incompletos, inconsistentes, ou muito antigos ao ponto de não

representar mais o estado de uma entidade. Portanto, a qualidade da informação de contexto é um elemento importante para o fornecimento de serviços ubíquos consistentes e eficazes. Segundo Buchholz et al. [3] a Qualidade de Contexto (*Quality of Context* - QoC) tem por objetivo descrever a qualidade da informação que é utilizada como contexto e para isto, define os seguintes parâmetros: *precision*, *probability of correctness*, *trust-worthiness*, *resolution*, *up-to-dateness*. Informações de contexto com qualidade exercem um papel importante para o funcionamento de aplicações cientes de contexto pois impactam diretamente em seu comportamento.

A utilização de informações contextuais com baixa qualidade pode resultar no mal funcionamento das aplicações cientes de contexto causado por problemas de precisão, acurácia, atrasos na atualização dos dados e isso tem grande impacto em sua experiência de uso.

Os chamados Serviços Baseados em Localização (*Location Based Services* - LBS) englobam aplicações cientes de contexto que dependem de informações contextuais a respeito da localização do usuário para fornecer serviços, sejam eles para áreas como saúde, educação, segurança, vendas ou navegação, que objetiva guiar um usuário por meio de instruções que o mostrem como ir de sua localização atual até um determinado local pretendido [4].

Em sistemas de navegação, uma tecnologia amplamente utilizada é o Sistema de Posicionamento Global (*Global Positioning System* - GPS). Diversos dispositivos disponíveis no mercado, como os *smartphones*, por exemplo, possuem módulo receptor de sinais de GPS embutido e esta tecnologia tem se mostrado bastante efetiva, com capacidade de obter localização com uma boa precisão. Contudo, a eficiência e boa precisão do GPS limita-se a seu uso em ambientes externos (*outdoor*), visto que em ambientes internos (*indoor*) seus sinais sofrem atenuação e são bloqueados por obstáculos como edifícios ou materiais de construção [5]. Além disso, a precisão exigida para o correto funcionamento de serviços de localização em ambientes internos é maior que em ambientes externos [6]. Um outro problema é que o GPS não é capaz de identificar a altura (ou andar) em que o usuário se encontra.

Diversas alternativas ao uso do GPS em ambientes indoor têm sido estudadas e propostas na literatura [7] [8] [9]. O INavigS [10], cenário deste estudo, é uma infraestrutura de software sensível ao contexto desenvolvida no âmbito de

um projeto de pesquisa da Universidade Federal do Maranhão que objetiva fornecer serviços de navegação em ambientes internos utilizando a comunicação entre beacons, pequenos dispositivos alimentados por bateria que transmitem informações através da tecnologia *Bluetooth Low Energy* (BLE), e *smartphones* para obter a localização do usuário.

A infraestrutura INavigS oferece serviços como o cálculo de rotas, assistência de navegação durante o percurso, geocodificação, além de disponibilizar todos os seus serviços em forma de *Application Programing Interface* (API) *Android* e *JavaScript* possibilitando a implementação de novas aplicações de navegação *indoor*. Apesar de desempenhar estes serviços, foram identificados problemas referentes a Qualidade de Contexto na infraestrutura:

- Os dados de contexto utilizados para definir a localização do usuário possuem baixa precisão, que pode ocasionar o cálculo errado de rotas de instruções de navegação, por ser incapaz de definir com exatidão o espaço em que o usuário se encontra;
- Inexistência de informações de contexto sobre a orientação do usuário. As instruções de navegação fornecidas precisam indicar a direção que o usuário deve seguir (e.g., siga em frente, vire à direita, vire à esquerda etc.), e para isso é preciso obter informações de contexto mais completas sobre a orientação do usuário.
- As instruções de navegação estão sujeitas a sofrer um alto tempo de atualização. Isto afeta diretamente a experiência de uso durante a navegação já que em um curto de intervalo de tempo, a localização do usuário pode mudar e se tais instruções não acompanharem esta mudança frequentemente estarão incorretas.

Foram identificados parâmetros de QoC existentes na literatura que podem ser aplicados para a solução da problemática proposta, que afeta o funcionamento da infraestrutura INavigS no que diz respeito à experiência de uso de suas aplicações. Os parâmetros são: *precision*, *completeness* e *refresh rate*.

1.1 Objetivos

O objetivo geral desta pesquisa é propor soluções através da aplicação de parâmetros de qualidade de contexto (*precision*, *completeness* e *refresh rate*) ao cenário da infraestrutura INavigS.

A pesquisa tem como objetivos específicos:

- Investigar o estado da arte em relação a Qualidade de Contexto;
- Investigar o estado da arte em relação a Técnicas de Localização *Indoor*;
- Modelar e adaptar as técnicas identificadas para a problemática em questão;
- Implementar as Técnicas de Localização *Indoor* e Qualidade de Contexto na Infraestrutura INavigS
- Avaliar se o uso das técnicas propostas melhoraram a experiência de uso do INavigS em termos de Qualidade de Contexto

1.2 Estrutura da Dissertação

Esta dissertação está organizada como segue:

- O Capítulo 2 apresenta os principais conceitos, métodos e tecnologias que servem de base fundamental para o desenvolvimento do trabalho.
- O Capítulo 3 discute trabalhos relacionados relevantes voltados para a área de navegação *indoor*, descrevendo sua arquitetura e as principais funcionalidades de cada um deles.
- O Capítulo 4 descreve os métodos utilizados, seus principais aspectos de implementação e a arquitetura INavigS após a implementação das soluções propostas.
- O Capítulo 5 apresenta os resultados obtidos com a avaliação dos métodos propostos neste trabalho.

- O Capítulo 6 apresenta as conclusões obtidas a partir desta pesquisa e trabalhos futuros.

2 Fundamentação Teórica

Neste capítulo será apresentada a fundamentação a respeito de temas relacionados ao contexto em que este trabalho está inserido. Serão mostrados conceitos, classificações e características sobre Computação Ubíqua, Computação Sensível ao Contexto, Qualidade de Contexto, a Infraestrutura INavigS e Técnicas de Localização *Indoor*.

2.1 Computação Ubíqua e Sensível ao Contexto

No ano de 1991, Weiser [1] imaginou um futuro em que as formas de interação entre humanos e computadores seriam diferentes das praticadas naquela época, e de tão naturais, as pessoas fariam uso de dispositivos computacionais de maneira quase imperceptível. O autor denominou este paradigma de Computação Ubíqua e quase três décadas depois é possível perceber a concretização dessa ideia quando observamos os avanços da computação móvel e tecnologias de comunicação sem fio.

Os *smartphones* passaram a fazer parte da vida das pessoas. Estes dispositivos oferecem diferentes formas de interação como voz ou gestos, capacidade computacional e capaz de processar e executar tarefas cada vez mais complexas. Além disso, objetos do dia-a-dia, como lâmpadas, aparelhos de ar-condicionado, televisão, entre outros, também ganharam a capacidade de se comunicar com outros dispositivos via tecnologias de redes de computadores.

Por ser um tema de pesquisa abrangente, o termo Computação Ubíqua envolve como subáreas a Computação Móvel e a Computação Pervasiva, que por vezes são considerados sinônimos mas apresentam algumas diferenças em seus conceitos.

O termo Computação Pervasiva indica a presença de inúmeros dispositivos computacionais distribuídos em ambientes comumente frequentados pelas pessoas, e está mais relacionado à onipresença da computação na vida cotidiana. Sendo assim, o

uso da computação passa a ser algo transparente, em que o usuário utiliza seus serviços e aplicações de forma imperceptível [11].

Já a Computação Móvel está relacionada a sistemas distribuídos em diferentes dispositivos que trocam informações entre si. É caracterizada pelo uso de hardware leve e portátil além de tecnologias de redes sem fio, que possibilitam a mobilidade. Assim, aplicações móveis oferecem continuidade em seu uso mesmo que o usuário se mova por diferentes ambientes físicos [12].

Por fim, a Computação Ubíqua envolve a integração da mobilidade oferecida pela computação móvel com as características da computação pervasiva, possibilitando ainda que as aplicações adaptem seu comportamento de acordo com o ambiente em que estão inseridas, mesmo que este mude. A Figura 2.1 ilustra a ligação entre os três termos.

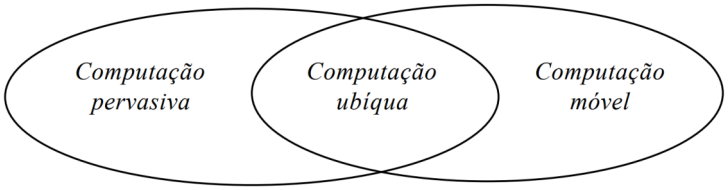

Figura 2.1: Relação entre os conceitos de Computação Ubíqua, Pervasiva e Móvel [13].

A sensibilidade ao contexto é uma característica das aplicações da Computação Ubíqua. Uma das primeiras aplicações da Computação Ciente ou Sensível ao Contexto foi o "Active Badge Location System", apresentado em 1992 por Want et al. [14] que tinha como objetivo fornecer a localização de pessoas dentro de um prédio composto por vários escritórios. Eram utilizados crachás que transmitiam sinais para fornecer informações sobre a localização de seus usuários.

Diversos autores têm explorado ao longo dos anos a definição para a chamada Computação Ciente de Contexto. Para Schilit et al. [15] significa a habilidade de uma aplicação ou dispositivo computacional de detectar, interpretar, agir e responder a aspectos do ambiente tais como localização, tempo, temperatura ou até mesmo o perfil do usuário. Em Theimer et al. [16] é vista de uma perspectiva em que a aplicação examina o ambiente computacional e reage a mudanças dinâmicas tais como a localização do usuário, o conjunto de pessoas próximas e dispositivos acessíveis para adaptar seu comportamento. Segundo Salber et al. [17] a sensibilidade ao contexto

pode ser utilizada para automatizar um sistema de software, modificar sua interface e flexibilizar o uso de recursos computacionais.

Ainda de acordo com Salber et al. [17] aplicações sensíveis ao contexto se diferenciam das tradicionais porque a obtenção das informações vão além dos métodos tradicionais em que o usuário precisa explicitá-las para a aplicação via dispositivos de entrada como mouse, teclado ou telas sensíveis ao toque. Na Computação Sensível ao Contexto as aplicações obtém os dados por meio de novos métodos de interação, como voz ou gestos, por exemplo, ou ainda através de sensores.

Para Abowd et al. [2], contexto refere-se a qualquer informação que possa ser utilizada para caracterizar a situação de uma entidade. Pode-se entender como entidade: uma pessoa, um lugar, um objeto ou qualquer outra coisa que tenha um papel relevante para a interação entre o usuário e a aplicação, incluindo eles mesmos.

Inicialmente, era dito que o contexto representava apenas informações sobre a localização, o perfil de usuário e objetos nas proximidades, bem como, as mudanças no estado desses objetos [15]. Em Schmidt et al. [18] é afirmado que contexto é o conhecimento sobre o estado atual de um usuário ou dispositivo computacional incluindo a sua localização e a situação do ambiente ao seu redor.

Com o intuito de simplificar o entendimento do que viria a ser contexto, alguns autores apresentaram formas de classificá-los. Para Abowd et al. [2], o contexto se dividiria em quatro categorias: localização, identidade, atividade e tempo.

Entretanto, à medida em que novas aplicações sensíveis ao contexto foram surgindo, percebeu-se que o termo contexto deveria ser utilizado de maneira mais ampla, de forma que pudesse representar mais tipos de informação. Em Chen et al. [19], contexto foi teve seu significado ampliado e passou a ser classificado em quatro categorias:

- Contexto Computacional: São dados sobre o ambiente computacional em que a aplicação está sendo executada. Exemplos: conectividade de rede, largura de banda, recursos computacionais disponíveis etc.
- Contexto do Usuário: Se refere a dados sobre o usuário que está utilizando a aplicação. Exemplos: perfil do usuário, sua localização etc.

- Contexto Físico: Descreve o ambiente físico em que o usuário ou o sistema está inserido. Exemplos: luminosidade, som, temperatura etc.

- Contexto do Tempo: Dados extraídos de informações sobre o tempo atual. Exemplos: hora, dia da semana, ano, estação etc.

Para Jang et al. [20], até então todas as formas de modelar o contexto eram muito específicas e centradas na finalidade de cada serviço ou aplicação. O autor apresentou uma abordagem mais genérica e unificada que estava mais centrada no estado do usuário. Ele dividiu o contexto em 6 dimensões e as chamou de 5W1H: *who*, *where*, *when*, *what*, *why* e *how*.

A dimensão *who* indica quem está utilizando a aplicação e divide-se em informações sobre o usuário como seu nome, id e senha; e seu perfil, que indica suas preferencias ou relacionamentos pessoais, por exemplo. A dimensão *where* representa informações a respeito da sua localização. O momento em que um contexto está disponível ou que foi gerado é representado por *when* que indica sua validade por apenas um certo intervalo de tempo. A dimensão *why* representa o estado mental do usuário, isto é, suas intenções e emoções. Já a dimensão *what* indica o objeto o qual o usuário está interagindo ou voltando sua atenção. Por fim, *how* descreve o comportamento e sinais biológicos do usuário.

2.1.1 Arquitetura de aplicações sensíveis ao contexto

Os sistemas sensíveis ao contexto podem ser implementados de muitas formas [21]. A maneira como será desenvolvido depende de seus requisitos e condições. Alguns aspectos que devem ser observados são: a localização dos sensores e se a comunicação com eles deverá ser local ou remota; a quantidade de usuários esperada, isto é, se o sistema será utilizado por apenas uma ou por várias pessoas ao mesmo tempo; os recursos disponíveis, que podem ser computadores com alto poder de processamento ou dispositivos móveis com capacidades limitadas; ou ainda a possibilidade de extensão do sistema. Para Chen et al. [19] outro aspecto que deve ser observado é a forma como os dados de contexto serão obtidos, podendo ser através de:

Acesso Direto ao Sensor: Neste caso, o sistema obtém o dado acessando diretamente o sensor, sem a necessidade de nenhuma camada adicional para tratá-los. Não é recomendável para sistemas distribuídos, pois conexão direta ao sensor não fornece mecanismos de acesso concorrente, por exemplo.

Infraestrutura de Middleware: Consiste em utilizar métodos para separar detalhes de programação entre a coleta e o uso dos dados de contexto. A abordagem baseada em *middleware* introduz uma arquitetura que possibilita "esconder" detalhes de programação de baixo nível, permitindo ao programador de aplicações estar mais focado no uso do contexto.

Servidor de Contexto: Esta abordagem estende a arquitetura baseada em *middleware* incluindo um componente de gerenciamento de acesso aos sensores por múltiplos usuários. Os dados de contexto coletados são enviados para o chamado servidor de contexto e assim o acesso concorrente por vários clientes passa a ser possível. Esta abordagem transfere altas cargas de processamento para o servidor, livrando os clientes móveis desta tarefa, já que normalmente possuem limitações em suas capacidades de hardware.

Em Ailisto et al. [22] e Baldauf et al. [21] são apresentadas arquiteturas conceituais que separam a detecção e o consumo do contexto em etapas executadas por diferentes camadas, tornando os sistemas mais extensíveis e reutilizáveis. Ainda em Ailisto et al. [22] estas arquiteturas são divididas em 5 níveis: **Físico**: que é onde estão localizados os sensores e outros objetos que produzem dados brutos; **Camada de Dados**: que é a camada onde os dados brutos são processados e estão mais próximos de virarem informação; **Camada Semântica**: que transforma as informações obtidas pela camada anterior em informações significativas para que seja possível inferir o contexto; **Camada de Inferência**: que usa informações da Camada Semântica e regras de inferência para fazer suposições sobre o que o usuário ou um dispositivo está fazendo e que tipo de serviço ele pode querer fazer; Por fim, no último nível estão as **Aplicações** que irão consumir as informações contextuais obtidas a partir das demais camadas.

Na Figura 2.2 são apresentadas, em uma abordagem *bottom-up*, as cinco camadas presentes no modelo conceitual da arquitetura proposta por Baldauf et al. [21]. Na camada de sensores estão localizadas quaisquer que sejam as fontes de

dados capazes de obter informações sobre o ambiente, que possam vir a ser utilizadas como contexto. É importante notar, que apesar deste nome, o autor afirma que podem ser sensores ou qualquer outro objeto com esta finalidade. A segunda camada, de aquisição de dados, é responsável por capturar os dados de contexto brutos, consultando-os nos sensores virtuais ou lógicos, presentes na primeira camada, através de *drivers* ou APIs e geralmente é reutilizável, escondendo detalhes de baixo nível de programação.

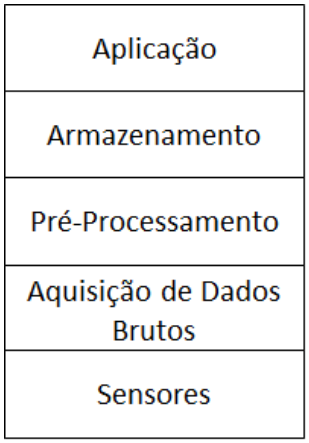

Figura 2.2: A arquitetura conceitual de uma aplicação ciente de contexto proposta por Baldauf et al. [21].

A terceira camada, chamada de pré-processamento, nem sempre é implementada em sistemas sensíveis ao contexto, mas pode ser útil quando os dados brutos forem muito "granulados", isto é, originados por múltiplas fontes. Esta camada é responsável por raciocinar e interpretar as informações de contexto, tornando-as mais apropriadas para serem consumidas. A quarta camada, organiza e armazena os dados coletados e os oferece por meio de uma interface pública para as aplicações cliente. Por fim, a quinta e última camada é a responsável por prover serviços e/ou informações aos usuários. É onde estão presentes as aplicações que consomem o contexto.

2.1.2 Os Requisitos de uma Aplicação Sensível ao Contexto

Durante o desenvolvimento de uma aplicação ciente de contexto, alguns requisitos são essenciais e precisam ser atendidos para que esta funcione adequadamente. São eles: Definição do Escopo, Aquisição, Representação, Processamento, Armazenamento, Uso e Compartilhamento.

É necessário realizar a **definição do escopo** para que antes de tudo se identifique quais entidades e informações contextuais serão suportadas pela aplicação. O escopo está diretamente relacionado aos seus requisitos e funcionalidades. Esta é uma tarefa difícil pois em muitos casos a quantidade de informações para definir o contexto de uma entidade é ilimitado [23].

A **aquisição** é o processo de capturar dados que serão posteriormente transformados em informação de contexto a partir de diversas fontes, sejam elas sensores físicos, que capturam dados sobre o ambiente físico, sensores virtuais que detectam o estado atual do ambiente computacional ou mesmo por entrada explícita, que é quando o próprio usuário fornece o dado [24].

Segundo Baldauf et al. [21] é necessário armazenar os dados de contexto em uma forma de **representação** que os torne processáveis. Para Vieira et al. [25] este é um desafio a ser enfrentado no desenvolvimento de sistemas sensíveis ao contexto. De acordo com Raychoudhury et al. [26] podem ser utilizadas diferentes técnicas para representar e modelar os dados de contexto, seja utilizando orientação a objetos, modelos gráficos, ontologias, grafos contextuais, mapas de tópicos entre outras.

O **processamento** é um dos requisitos essenciais em aplicações sensíveis ao contexto, pois segundo Vieira et al. [25] é nesta etapa que é possível produzir informações realmente significativas a partir dos dados brutos. Pode ser entendido como um conjunto de etapas que se dividem em: *raciocínio*, fase em que é feita a inferência dos dados de baixo nível, transformando-os em informações de alto nível; *transformação*, que realiza a "tradução" das informações para outros formatos que possa ser exigido por determinada plataforma, por exemplo; *combinação* que realiza a fusão de diferentes dados para gerar novas informações; *tratamento de incertezas* que trata de problemas relacionados a inconsistência, ambiguidade e integridade das informações.

Um outro requisito importante é o **armazenamento**, que permite que as informações contextuais possam ser solicitadas posteriormente, mesmo que já tenham sido utilizadas, com a finalidade de resolver conflitos ou mesmo para inferir novas informações, por exemplo [25].

A **utilização** é a etapa em que as informações de contexto serão usadas para guiar as ações das aplicações e Vieira et al. [25] a divide em: percepção, que indica

a utilização das informações de contexto para oferecer recomendações ao usuário; assistência que diz respeito a auxiliar o usuário na execução de tarefas apresentando dicas, por exemplo; e adaptação: que significa modificar a ação das aplicações de acordo com as informações de contexto inferidas.

A Qualidade de Contexto (*Quality of Context* - QoC) é um outro requisito essencial para aplicações sensíveis ao contexto. Por ser um dos temas centrais deste trabalho, será abordado em maiores detalhes na próxima seção.

2.2 Qualidade de Contexto (QoC)

Como dito anteriormente, o contexto pode ser obtido a partir de diversas fontes, tais como sensores, sendo eles físicos ou virtuais, ou ainda explicitados pelo usuário da aplicação. Os dados brutos sobre o ambiente são capturados, transformados em informação de contexto para então serem consumidos pelas aplicações. Estes dados nem sempre são confiáveis, podendo apresentar erros e inconsistências. Por exemplo, os sensores de GPS - *Global Positioning System*, que capturam dados de contexto sobre localização, pode ter sua precisão afetada de acordo com a intensidade dos sinais recebidos pelos satélites. A taxa de erro pode variar de metros a quilômetros, dependendo do ambiente [27].

Segundo Henricksen et al. [28], os erros em dados de contexto capturados por meio de sensores físicos são causados principalmente por indisponibilidade ou defeito do sensor, fatores do ambiente que alterem seu funcionamento (interferências ou alta temperatura, por exemplo), ou até mesmo pela forma como os dados são interpretados pelos algoritmos de inferência de contexto utilizados. Além disso, a utilização de múltiplos sensores pode resultar em inconsistências, já que diferentes sensores podem detectar divergentes valores para a medição de um mesmo dado.

Em Buchholz et al. [3] a Qualidade de Contexto (QoC) é definida como qualquer informação que seja capaz de descrever a qualidade de uma informação que é utilizada como contexto. O autor ainda explica a diferença entre os termos "Qualidade de Serviço" (QoS), "Qualidade do Dispositivo" (QoD) e "Qualidade de Contexto" (QoC). A QoS está relacionada à qualidade de execução de algum serviço, como a taxa de transmissão de dados por uma rede, por exemplo. Já a QoD relaciona-

se com a capacidade e características técnicas de um dispositivo. A QoC refere-se exclusivamente às características das informações utilizadas como contexto, em termos de sua qualidade.

Para Krause et al. [29], a QoC é qualquer informação inerente que descreve uma informação de contexto e que pode ser usada para determinar o nível de perfeição da informação para uma aplicação específica. Em uma abordagem mais recente, Manzoor et al. [30] afirmou que a QoC tem sido insatisfatória desde que se iniciou a pesquisa em sistemas sensíveis ao contexto. O autor define QoC como a identificação do grau de conformidade do contexto coletado por sensores com a situação real vigente no ambiente de acordo com os requisitos de um consumidor de contexto.

Exemplificando, em uma situação em que se tem dois provedores de contexto e um deles consegue obter a localização do usuário por meio de informações de redes *Global System for Mobile Communications* (GSM) com a precisão a nível de cidade e o outro utiliza tecnologia de GPS que consegue obter uma precisão a nível de rua. Para uma aplicação que irá consumir esses dados de contexto com o objetivo de apresentar pontos turísticos da cidade em que o usuário se encontra, tais dados podem conter qualidade e um nível de precisão aceitável. Entretanto, para uma outra aplicação, que tem como finalidade apresentar pontos turísticos dentro de um museu, esses dados possuem qualidade e precisão insuficiente. Este exemplo indica que a qualidade de contexto deve ser medida de acordo com os requisitos do consumidor do contexto, e a maneira com que a informação será utilizada.

Ainda em Manzoor et al. [30] é apresentado um modelo para a definição do nível de QoC dividido em camadas, mostrado na Figura 2.3.

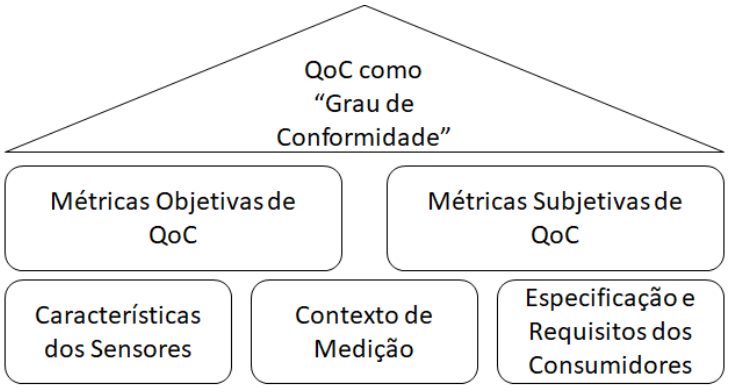

Figura 2.3: Modelo de definição do nível de QoC [30].

A primeira camada envolve as **Características dos Sensores** e demais fontes que capturam as informações de contexto e suas características que podem afetar a qualidade das informações; o **Contexto de Medição**, que indica informações relacionadas à situação específica de uma medição, como localização do sensor, tempo de medição etc; **Especificações e Requisitos dos Consumidores** que irão indicar as necessidades e exigências demandadas pelo consumidor de contexto.

A camada superior é derivada pelas informações da camada inferior com as informações sobre as características do sensor, contexto de medição, especificações e requisitos do consumidor de contexto. Divide-se em **Métricas Objetivas** e **Subjetivas**. As **Métricas Objetivas de QoC** são definidas independentemente dos requisitos dos consumidores de contexto, e mostram se as informações coletadas são livres de erros e adequadas para o uso. As **Métricas Subjetivas de QoC** são definidas a partir dos requisitos dos consumidores de contexto.

A importância e as razões para se considerar o uso de QoC em aplicações sensíveis ao contexto são apresentadas em Buchholz et al. [3] e Sheikh et al. [31]. A primeira delas é que, já que estas aplicações precisam das informações de contexto para executar seus serviços, a imperfeição teria um impacto significativo em sua usabilidade. A segunda é que a QoC pode ser usada como um indicador para a seleção de provedores de contexto, caso se queira escolher um provedor que ofereça dados com mais QoC, por exemplo. Por fim, a qualidade de contexto pode melhorar a experiência de uso e privacidade.

Para que se possa analisar o nível de qualidade das informações contextuais, existem os chamados indicadores ou parâmetros de QoC. A seguir serão abordados os vários parâmetros de QoC e os seus respectivos significados.

Parâmetros de Qualidade de Contexto

Os parâmetros de QoC são informações associadas aos dados de contexto que têm por objetivo identificar sua qualidade. O trabalho de Nazario et al. [32] faz uma ampla análise da literatura sobre as diversas nomenclaturas e significados dos parâmetros definidos pelos diferentes autores da área. Com este trabalho pôde-se concluir que não há um consenso para a definição dos termos e nomenclaturas utilizadas para definir cada um deles.

Os parâmetros considerados por Buchholz et al. [3] são *precision*, *probability of correctness*, *trust-worthiness*, *resolution* e *up-to-dateness*. No trabalho de Kim et al. [33] foram propostos os parâmetros: *accuracy*, *completeness*, *representational consistency*, *access security* e *up-to-dateness*. Para Sheikh et al. [31] os parâmetros de QoC são: *precision*, *freshness*, *spatial resolution* e *probability of correctness*.

A seguir serão apresentados os principais indicadores de QoC encontrados na literatura.

- *Accuracy* - Este parâmetro descreve o quão exatamente a informação representa a realidade [3]. Em Manzoor et al. [30] é dito que este parâmetro indica o quão próximo é medida, calculada ou coletada uma informação de contexto em relação à situação atual verdadeira do mundo real. Para Kim et al. [34], *accuracy* é o grau em que uma informação de contexto está correta e confiável. O autor afirma que é difícil mensurar o correto valor da informação, portanto utiliza a estimativa de um intervalo de valores confiáveis através de métodos estatísticos.

- *Precision* - O parâmetro *precision* é descrito por Sheikh et al. [31] como a "granularidade", isto é, o nível de detalhamento com que uma informação de contexto consegue descrever uma situação do mundo real. Já para Dustdar et al. [35] este parâmetro indica o grau de exatidão de uma medição, ou seja, refere-se à capacidade de um sensor retornar um mesmo valor ao realizar uma determinada quantidade de leituras sobre o contexto a partir de uma mesma circunstância. O significado deste parâmetro normalmente cofunde-se com o de *accuracy*, e Dustdar et al. [35] afirma que a diferença é que enquanto *accuracy* de um sensor apresenta o quão próximo o valor de uma medição está para o valor real, precisão apresenta o quão próximos estão os valores de sucessivas leituras do sensor em uma mesma circunstância.

- *Up-to-dateness* - Descreve a idade de uma informação contextual [3] [31]. Para Kim et al. [33] e Wang et al. [36] este parâmetro descreve até que ponto uma informação contextual está atualizada para uma tarefa em questão. Este mesmo parâmetro é chamado de *freshness* em Manzoor et al. [30] e representa o quão novo é um dado de contexto.

- *Resolution* - Para Dustdar et al. [35] este parâmetro indica o nível de detalhamento com que um dado pode ser coletado. Em Sheikh et al. [31] este parâmetro divide-se em temporal, que indica o período de tempo em que um amostra da informação é válida; e espacial, que refere-se a granularidade com que se pode expressar a área física na qual uma informação de contexto é aplicável. Para Buchholz et al. [3] este parâmetro é tido como o nível de "granularidade" de uma informação de contexto.

- *Probability of Correctness* - Este parâmetro estima a probabilidade de uma informação contextual estar errada, devido a falha em sensores, por exemplo [3]. Para Huebscher et al. [37] este parâmetro pode ter variação em seu valor caso sejam utilizados sensores de tipos diferentes, como a utilização de uma câmera e um sensor para detectar presença, por exemplo.

- *Trust-worthiness* - Para Buchholz et al. [3] este parâmetro também refere-se à probabilidade de uma informação estar correta, entretanto é utilizado pelo provedor de contexto para avaliar o nível de confiança dos dados obtidos por um sensor, por exemplo. Para isto, é verificado se os dados anteriormente recebidos por um sensor estavam corretos ou a sua porcentagem de erros.

- *Completeness* - Este parâmetro indica a quantidade de atributos de contexto fornecida [33]. Em Manzoor et al. [30] é o indicativo de que todos os aspectos do ambiente necessários foram mostrados pelas informações de contexto obtidas.

2.3 *Mobile Hub* (M-Hub)

O *Mobile Hub* (M-Hub) [38] foi desenvolvido no âmbito do projeto ContextNet[1], por pesquisadores do *Laboratory For Advanced Collaboration* (LAC) da Pontifícia Universidade Católica do Rio de Janeiro (PUC-Rio) em conjunto com pesquisadores do Laboratório de Sistemas Distribuídos Inteligentes (LSDi) da Universidade Federal do Maranhão (UFMA). É um *middleware* de propósito geral voltado para dispositivos móveis, que fornece a descoberta, conexão e aquisição de dados originados por outros tipos de objetos móveis (*Mobile Objects* - M-OBJs) que possuem apenas tecnologias de comunicação de curto alcance via *Wired Personal Area*

[1] `http://wiki.lac.inf.puc-rio.br/doku.php`

Network (WPAN) como *Bluetooth Classic*, *Bluetooth Low Energy* (BLE), ZigBee, NFC, ANT+ etc.

Desta forma, dispositivos móveis agem como *gateways*, direcionando os dados recebidos pelos *M-Objs* (e.g., sensores, atuadores, relógios inteligentes, veículos etc.) para a *internet* por meio de conexões como Wi-Fi ou 3G/4G, para que sejam processados por nós estacionários.

O M-Hub oferece suporte a dispositivos móveis como os *smartphones*, fortes candidatos a atuarem como *gateways* pois ganharam popularidade e contam com diversas tecnologias embutidas, desde sensores, que capturam dados do ambiente, a interfaces de comunicação diversas, contando com tecnologias como WLAN e WPAN.

Entre os serviços fornecidos pelo M-Hub estão:

- **Descoberta de M-Objs próximos:** oportunisticamente o M-Hub detecta os objetos móveis inteligentes próximos que estejam transmitindo seu ID e lista de serviços oferecidos, com a possibilidade de armazenar estas informações em uma base de dados.
- **Conexão com os M-Objs próximos:** dependendo da tecnologia de comunicação WPAN utilizada, existe a necessidade de estabelecimento de conexão entre o *smartphone* e o M-Obj detectado, para que seja possível receber os dados;
- **Controle e Gerenciamento dos M-Objs:** alguns M-Objs oferecem a possibilidade de serem configurados por meio de parâmetros e comandos enviados via WPAN e o M-Hub fornece serviços para enviar estes comandos aos sensores com conexão estabelecida;
- **Pré-processamento e Transcodificação de Dados de Sensores:** antes de enviar os dados pode ser preciso certificar-se de que eles obedecem a um formato que possa ser legível pelos consumidores de contexto e o M-Hub é capaz de realizar este pré-processamento;
- **Carregamento Dinâmico de módulos de sensores:** o M-Hub oferece mecanismos para a implantação e gerenciamento de módulos específicos de sensores, visto que nem sempre é possível ter módulos referentes a todos os sensores presentes no ambiente em que o usuário se encontra;

- **Privacidade:** O M-Hub conta com uma funcionalidade para filtrar dados de contexto que não possam ser enviados livremente pela rede. Para isto, os dados podem ser marcados como "privativos" e passam a ser acessíveis apenas para visualização local pelo M-Hub View;

- **Processamento de acordo com o nível de Energia:** O M-Hub oferece ainda um mecanismo que controla o uso dos recursos pelos serviços do *middleware* de acordo com o nível de bateria disponível no dispositivo móvel, por exemplo, reduzindo a periodicidade com que os dados são enviados pela rede se o nível de bateria chegar a um determinado nível.

A Figura 2.4 mostra a arquitetura do M-Hub e seus componentes. São eles:

- **Short-Range Sensor, Presence and Actuation Service (S2PA):** o serviço oferecido por este componente é realizar a descoberta, conexão e a comunicação com os M-Objs próximos;

- **Location Service:** é responsável por monitorar a localização atual do dispositivo que executa o M-Hub associando-a a todos os dados coletados pelo dispositivo móvel e publicados pelo M-Hub. O objetivo é fornecer informações sobre a localização do usuário e objetos inteligentes próximos;

- **Connection Service:** é responsável por manter uma conexão entre o gateway bem como transmitir dados para a nuvem através do protocolo *Mobile Reliable UDP* (MR-UDP);

- **Energy Manager:** este componente verifica periodicamente o nível de bateria do dispositivo móvel e o classifica em alto, médio ou baixo. Com isto, ajusta a frequência com que os serviços do M-Hub funcionarão, como a busca por sensores, execução do *location service* para verificar a localização, envio de dados ao *gateway* etc. Seu objetivo é diminuir o consumo de bateria do dispositivo.

- **Mobile Event Processing Agent Service**: utiliza a tecnologia de um motor de inferência de dados chamado Esper CEP (Complex Event Processing) para avaliar fluxo de dados dos sensores à procura de correlação entre eles, como o tempo, valor ou ordem de ocorrência.

- **Mobile Client Adaptation Service:** permite a extensão do *middleware* através da implantação de novos módulos de sensores ou funcionalidades gerenciando o ciclo de vida do código Java destes módulos recebidos dinamicamente.

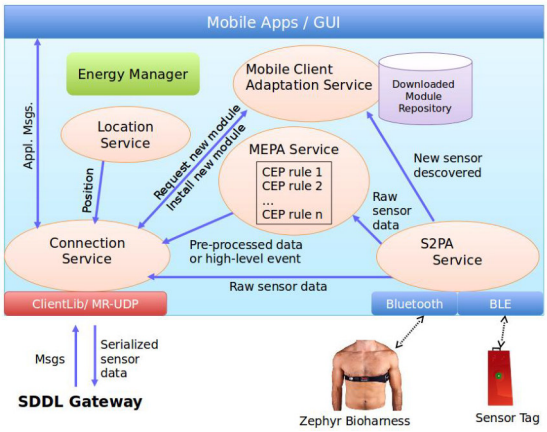

Figura 2.4: Arquitetura do *Middleware* M-HUB. [38]

O componente *Scalable Data Distribution Layer* (SDDL) é um *middleware* de comunicação que é usado para conectar e forncer comunicação entre nós estacionários, presentes em redes com fio, a nós móveis com conexão sem fio tais como 3G/4G ou Wi-Fi [38]. Os nós estacionários podem ser os *gateways*, utilizados para prover a comunicação com os nós móveis, ou nós de processamento de informações de dados de contexto.

O SDDL utiliza dois protocolos de comunicação: o *Data Distribution Service* (DDS) que provê a comunicação com fio entre os componentes do núcleo SDDL e o *Mobile Realiable UDP* (MR-UDP), usado para a interação entre o núcleo da rede e os nós móveis. Este protocolo tem características semelhantes às do TCP e foi personalizado para lidar com conectividade intermitente e outras características comuns em ambientes que envolvem redes sem fio e nós móveis, como alterações no endereço IP e nas interfaces de rede.

A infraestrutura do INavigS, cenário deste trabalho, utiliza alguns do serviços e componentes oferecidos pelo SDDL e M-Hub e será descrita na próxima seção.

2.4 A infraestrutura INavigS

O *Indoor Navigation System* (INavigS) é uma infraestrutura de *software* com sensibilidade ao contexto, que suporta dispositivos móveis e objetiva fornecer serviços de navegação *indoor*.

O INavigS utiliza dados de contexto tais como a localização, o perfil do usuário e o nível de bateria do dispositivo móvel para ajustar o seu comportamento. Os dados de contexto sobre localização do usuário são utilizados para traçar rotas de navegação, guiando o usuário desde o seu local atual até um determinado ponto pretendido. A partir de informações sobre o perfil do usuário são fornecidas instruções de maneira personalizada. O nível de bateria do dispositivo móvel ajusta a frequência com que mensagens entre os componentes da arquitetura serão trocadas.

A principal motivação para o desenvolvimento desta infraestrutura foi suprir a necessidade de serviços de navegação em ambientes internos — que são lugares em que o *Global Positioning System* (GPS), uma tecnologia amplamente utilizada e eficaz não funciona adequadamente — e facilitar a implementação de aplicações ubíquas que necessitam dos seus serviços, que são:

- Identificação da localização do usuário no ambiente interno;
- Geocodificação, que é a conversão de espaços internos (e.g., sala de reunião, biblioteca etc.) em coordenadas geográficas (e.g., Latitude: -2.559355, Longitude: -44.307985);
- Cálculo de rotas e fornecimento de informações sobre possíveis obstáculos durante o percursos, como escadas, rampas ou elevadores;
- Assistência e fornecimento de instruções enquanto o usuário percorre a rota (e.g., siga em frente, vire à direita etc.)

Uma outra característica do INavigS, é que estes serviços podem ser personalizados de acordo com o perfil do usuário. Por exemplo, se o usuário for um portador de necessidades especiais, o cálculo da rota evitará caminhos que necessitem que o usuário tenha que passar por escadas.

Em sua parte física, a infraestrutura conta com o uso de *smartphones*, que são cada vez mais populares e de preços acessíveis, além de dispositivos

denominados *beacons* BLE que recebem este nome devido à presença de uma interface de comunicação compatível com o BLE. Desta forma, emitem sinais que podem ser recebidos por qualquer *smartphone* ou dispositivo móvel habilitado para esta tecnologia [39]. A partir das mensagens recebidas, o *smartphone* é capaz de identificar o quão próximo ou distante está de cada um dos *beacons*, o que significa que diferentes ações podem ser realizadas dependendo se o dispositivo móvel estiver dentro de, por exemplo, um raio de 5, 10 ou 50 metros. Diversos *beacons* podem ser detectados simultaneamente pelo *smartphone* e, calculando sua distância relativa a cada um deles, é possível deduzir a sua localização.

Um requisito exigido para o funcionamento da infraestrutura é a necessidade de conectividade com módulos distribuídos em diferentes computadores ou *clusters* durante todo o percurso, para que processamentos mais complexos sejam feitos externamente ao dispositivo móvel.

Arquitetura do INavigS

A infraestrutura INavigS divide-se em duas camadas, e é composto por quatro componentes. Cada um deles é responsável por diferentes funções, que serão apresentadas a seguir. A Figura 2.5 mostra como estão organizadas as camadas e os componentes do INavigS.

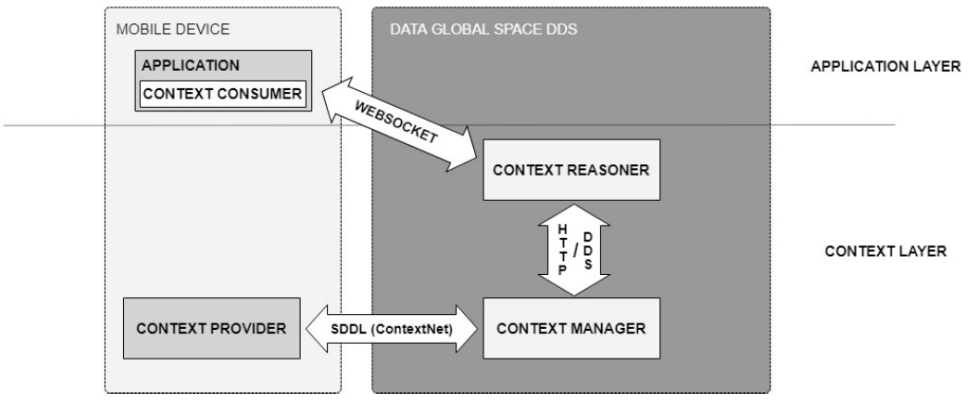

Figura 2.5: Arquitetura do *INavigS* [10].

A primeira camada, denominada ***Context Layer***, é responsável pela captura, gerenciamento e processamento do contexto. Entre seus componentes estão o *Context Provider*, *Context Manager* e *Context Reasoner*.

O *Context Provider* é o componente responsável por capturar, representar e realizar o pré-processamento do contexto. A implementação deste componente faz uso do *middleware* M-Hub apresentado anteriormente na Seção 2.3. São utilizados os serviços do M-Hub: *S2PA Service*, *MEPA Service* e *Connection Service*.

No escopo do INavigS, a principal informação de contexto a ser obtida é a localização do usuário no espaço *indoor*. Logo, esta é obtida a partir do registro do encontro entre o *smartphone* do usuário e os *beacons*.

Considera-se a posição do usuário como sendo a mesma do *beacon* com o maior valor para o parâmetro *Received Signal Strength Indicator* (RSSI) medido. O RSSI indica a força com que um sinal é recebido por um dispositivo. Quanto mais próximo o *beacon* estiver do dispositivo, maior será o RSSI recebido, e quanto mais distante, o valor deste parâmetro tende a diminuir. Este é o método de posicionamento baseado em Proximidade. Métodos de Posicionamento *Indoor* serão abordados em mais detalhes na Seção 2.5.

Os *beacons* utilizam a tecnologia de transmissão *Bluetooth Low Energy* (BLE) à qual a maioria dos *smartphones* atuais suportam. Assim, durante o uso do INavigS, o *smartphone* do usuário realiza o escaneamento de dispositivos BLE próximos, em busca dos *beacons* que fazem parte do INavigS. O *S2PA Service* do M-Hub realiza esta tarefa de escaneamento, registrando a descoberta de dispositivos. O encontro de cada dispositivo registrado é representado por uma classe Java com os atributos: UUID, que é um identificador único e o valor do RSSI. O *Energy Manager* controla a periodicidade com que o escaneamento é feito de acordo com o nível de bateria do dispositivo móvel, que se divide em três níveis: *LOW*, *MEDIUM* e *HIGH*.

O componente *Context Manager* realiza o armazenamento de dados que serão utilizados para inferir informações contextuais, além de realizar o processamento de consultas a esses dados. Este componente é implementado com o uso do HORYS[2], um software implementado em linguagem Java e desenvolvido dentro do escopo do projeto Hospital 4.0 da PUC-Rio. Este componente registra encontros entre dispositivos móveis (e.g., smartphones) que executam o M-Hub e M-OBJs. As informações ficam armazenadas em um banco de dados não-relacional (NoSQL). No cenário do INavigS, os M-OBJs são os *beacons* BLE. O *middleware* SDDL discutido na

[2] `http://interscity.org/software/contextnet/`

Seção 2.3 é utilizado para realizar a transferência dos dados do *Context Provider* para o *Context Manager*.

O *Context Reasoner* por sua vez, consulta periodicamente o *Context Manager* e utiliza os dados obtidos para realizar a inferência sobre o contexto e a produção de informações com maior valor agregado. A comunicação entre o *Context Reasoner* e o *Contex Manager* pode se dar por meio de requisições via protocolo DDS ou via serviços web (HTTP).

A conversão das informações sobre os *beacons* detectados em espaço semânticos e o fornecimento de instruções a serem seguidas para completar as rotas são tarefas deste componente, que disponibiliza suas informações processadas através do protocolo *Websocket*.

Na segunda camada do INavigS, denominada ***Application Layer***, residem as aplicações consumidoras dos serviços, citados anteriormente, que são oferecidos pela infraestrutura. Com o objetivo de apoiar a construção destas aplicações, foram desenvolvidas duas bibliotecas: uma em *JavaScript* para apoiar o desenvolvimento de aplicações Web e uma outra *Android* que é voltada para aplicações em dispositivos móveis. Estas bibliotecas fazem parte do componente *Context Consumer* e são apresentadas em [10].

No cenário de execução do INavigS, as ações se dão da seguinte forma:

1. Durante todo o processo, o *Context Provider*, que executa no *smartphone* do usuário, escaneia e envia os dados dos *beacons* BLE próximos para o *Context Manager* através do middleware SDDL;

2. Ao receber os dados, o *Context Manager* os armazenam em uma base de dados NoSQL e os disponibilizam para as camadas superiores;

3. Assim que recebe uma solicitação de assistência de rota, o *Context Reasoner* irá calcular a rota, levando em conta as características do perfil do usuário, e iniciará o monitoramente da sua posição no espaço interno;

4. Uma mensagem de confirmação é enviada à aplicação cliente, localizada na ***Application Layer***

5. O *Context Reasoner* passa a monitorar periodicamente o *Context Manager* para inferir a localização do usuário e verificar se ela mudou;

6. Caso haja mudança na localização do usuário, o *Context Reasoner* realizará o cálculo de uma nova instrução que será informada ao usuário;

7. Os passos se repetem até que o usuário da aplicação cliente solicite o cancelamento dos serviços de navegação ou chegue no local pretendido;

Conforme dito acima, a técnica de localização *indoor* utilizada pelo INavigS para determinar o posicionamento do usuário no espaço interno é baseada em Proximidade. Existem diversas outras técnicas na literatura, e elas serão abordadas em maiores detalhes na seção a seguir.

2.5 Os métodos de Localização *Indoor*

Entre os tipos de informações de contexto, a localização é uma das mais utilizadas. A partir do momento em que se sabe qual ambiente o usuário está, é possível inferir outras informações, como possíveis ações que ele está executando ou fornecer instruções sobre pontos de interesse e dispositivos próximos. A maioria dos *smartphones* disponíveis no mercado atualmente possuem GPS, tecnologia de localização já utilizada há décadas e que pode oferecer uma precisão que chega a metros.

Os *smartphones* habilitados com a tecnologia de GPS conseguem obter uma precisão com um raio de até 4,9 metros em céu aberto [40], no entanto, este nível de precisão piora à medida em que obstáculos, como prédios, pontes e árvores estão presentes. Outros fatores podem influenciar a precisão do GPS como condições climáticas, qualidade do sensor receptor dos sinais de satélite ou a presença de obstáculos que possam bloquear os sinais. Geralmente isto acontece em ambientes internos, como o interior prédios, onde os sinais são atenuados e bloqueados por paredes, telhados e materiais de construção [41].

Diante disto, pesquisadores têm desenvolvido métodos para serem utilizados como alternativas ao GPS para a obtenção de localização em ambientes internos. As tecnologias de comunicação sem fio aliadas aos dispositivos móveis

são vistas como promissoras e oferecem oportunidades para o desenvolvimento de pesquisas nesta área.

Entre as diversas tecnologias utilizadas, destacam-se o Wi-Fi, ZigBee, Bluetooth Classic e *Bluetooth Low Energy* (BLE), Radio-frequency Identification (RFID) e Infra-vermelho (IR). Entre elas, uma que tem ganhado atenção recentemente é o BLE, por oferecer uma série de vantagens, como o baixo consumo de energia, simplicidade de uso, possibilidade em se obter uma boa precisão e é compatível com diversos dispositivos móveis. Os *beacons*, são pequenos dispositivos que utilizam o BLE, que têm baixo custo e são facilmente instalados no ambiente, já que não precisam de infraestrutura adicional, e são alimentados por baterias que podem durar anos, dependendo de sua configuração [42].

As tecnologias citadas acima são combinadas com os diversos métodos de localização existentes, que se classificam em: Proximidade, Triangulação, Análise de Cena e *Dead Reckoning*. As subseções a seguir detalham cada um destes métodos expondo suas vantagens e desvantagens.

2.5.1 Localização baseada em Proximidade

A proximidade é uma técnica que determina a localização a partir da distância entre um dispositivo móvel e sensores fixados em áreas conhecidas [4]. Estes sensores podem ser *beacons* BLE, dispositivos RFID ou até mesmo *access point* de redes sem fio (Wi-Fi). A distância entre o dispositivo móvel e o sensor pode ser calculada, por exemplo, com base no nível de intensidade do sinal RSSI (RSSI), partindo-se do pressuposto de que quanto mais próximo se estiver do sensor, maior será a intensidade do sinal, e à medida em que se afasta dele, a intensidade diminui. A localização fica associada à área de atuação do sensor detectado. Se mais de um sensor for detectado, a área correspondente àquele com o maior nível de RSSI definirá a localização.

A Figura 2.6 mostra como a técnica funciona: E1, E2 e E3 são elementos que se deseja localizar. O retângulo tracejado representa a área de proximidade do sensor S. Assim, E2 e E3 estão localizados na área de atuação do sensor S1 e então sua localização será associada a ele. Já o elemento E3 está fora da área de atuação do sensor S1, visto que está mais próximo do sensor S2 e consequentemente o nível de RSSI recebido por S2 é maior. Portanto, sua localização ficará associada à área de atuação do sensor S2.

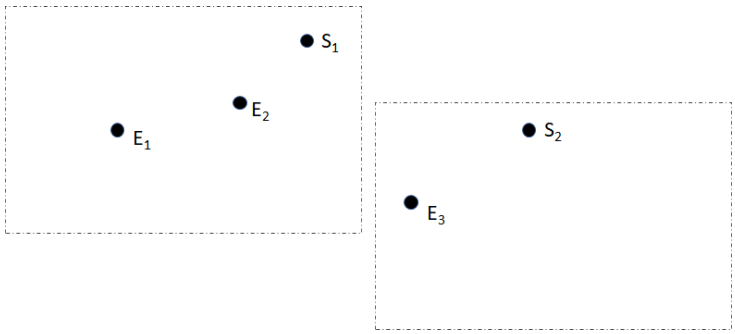

Figura 2.6: Representação da Técnica de Localização por Proximidade [4].

Este método tem como vantagem a sua facilidade de implementação, e sua principal desvantagem é o nível de precisão, que está diretamente associada à quantidade de sensores utilizados [43]. Não é possível definir a localização absoluta utilizando-se este método, como acontece em outras técnicas de localização [4].

2.5.2 Triangulação

A Triangulação é uma técnica que utiliza as propriedades geométricas dos triângulos para determinar a posição de um elemento. Em Farid et al. [44] ela é subdividida em dois tipos: Angulação e Lateração. No caso da Angulação, a localização é obtida através de cálculos que envolvem ângulos já a Lateração baseia-se na distância entre o elemento que se deseja localizar e um conjunto de pontos de referências, como sensores, por exemplo.

Alguns métodos podem ser utilizados para calcular a localização utilizando a triangulação: *Angle of Arrival* (AoA), *Time of Arrival* (ToA), *Time of Difference Arrival* (TDoA), e cálculo da distância por meio da medição da intensidade de sinais Received Strength Signal Indicator (RSSI).

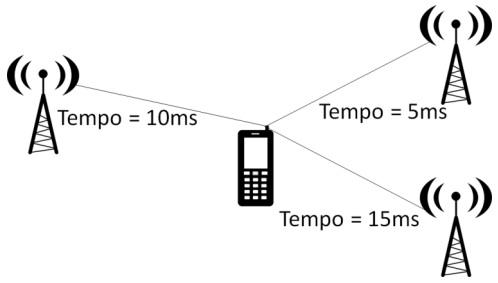

Figura 2.7: Representação da Técnica de Localização *Time of Arrival* (ToA) [44].

Segundo Gu et al. [4] o ToA é o método mais preciso, já que consegue filtrar problemas como o multi-percurso, ao mesmo tempo, sua implementação é a mais

complexa, devido a problemas como a sincronização dos relógios dos dispositivos que resultam em erros na estimativa do tempo. Consiste em utilizar o tempo de propagação do sinal emitido por um transmissor até ser recebido pelo receptor para computar a distância entre eles. A Figura 2.7 exemplifica o funcionamento do *Time of Arrival*.

O TDoA é um método também baseado no tempo de propagação dos sinais, uma vantagem em relação ao *Time of Arrival* (ToA) é que este não possui dependência com a sincronização de relógios. O que muda é que nesta será calculada a diferença do tempo de chegada de sinais transmitidos por diferentes tecnologias, como sinais sonoros e de rádio, por exemplo.

O método AoA determina o ângulo de incidência do sinal assim que ele chega no dispositivo a ser localizado em relação aos emissores, que estão em locais conhecidos. Este método funciona com apenas dois emissores, mas caso seja desejado uma melhor precisão, três ou mais também podem ser utilizados [44]. Este método necessita do uso de antenas direcionais ou matrizes de antenas e isto aumenta o seu custo de implantação. Em ambientes internos, a sua precisão é degradada por efeitos de *multipath* causados pela reflexão dos sinais em paredes e outros objetos. Além disso, um pré-requisito para que funcione corretamente é que haja linha de visão direta entre o receptor e os emissores. Por este motivo, o método AoA não é recomendável para ambientes *indoor*. A Figura 2.8 exemplifica como o *Angle of Arrival* funciona.

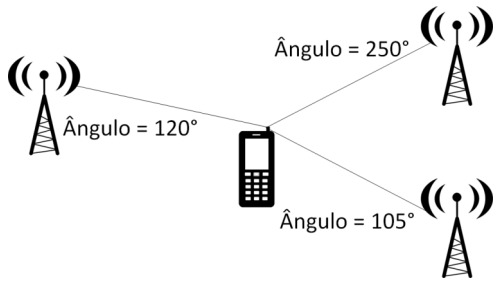

Figura 2.8: Representação da Técnica de Localização *Angle of Arrival* (AoA) [44].

O RSSI, como já brevemente explicado anteriormente, é diferente dos métodos apresentados anteriormente pois o parâmetro utilizado para determinar a localização é o decaimento da potência dos sinais de acordo com a distância. São utilizados modelos de propagação das ondas de sinais eletromagnéticos para traduzir os valores da potência de sinal em distância percorrida por eles [45].

Este método é utilizado na técnica de Lateração, que calcula a distância do elemento a ser localizado em relação a pontos de referência em posições conhecidas.

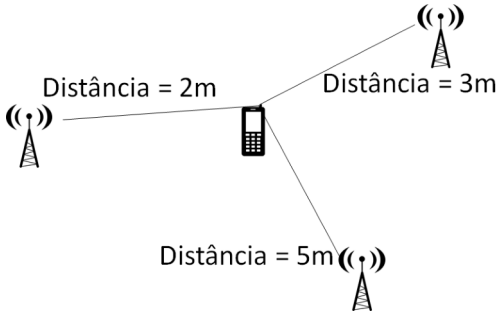

Figura 2.9: Representação da Técnica de Localização por Lateração [44].

Sua vantagem é a facilidade de implementação, mas também sofre problemas de precisão devido aos efeitos de *multipath* presente em ambientes *indoor*. A Figura 2.9 mostra um exemplo de como funciona a técnica de Lateração.

2.5.3 Análise de Cena

Esta técnica de localização *indoor*, também conhecida como Análise de Padrões, consiste na análise de um conjunto de dados do ambiente que sejam capazes de caracterizar uma localização [45]. Estas características podem ser obtidas a partir do processamento de imagens do ambiente (análise de imagens) ou de análise de características de sinais eletromagnéticos (*fingerprinting*).

A localização por análise de imagens estima a localização através de imagens recebidas por um ou múltiplos pontos [46]. Este método oferece conforto e eficiência aos usuários já que não há a necessidade de levar consigo dispositivos de rastreamento.

Uma ou várias câmeras são fixadas na área de rastreamento para obter imagens em tempo real. Estas imagens capturadas são comparadas com outras presentes em uma base de dados para que sejam feitas as estimativas de posição e identificar os objetos rastreados [47].

Diferente da análise de imagens, a técnica de *fingerprinting* analisa características dos sinais eletromagnéticos presentes no ambiente para obter a localização. O parâmetro comumente utilizado é o nível de intensidade (RSSI) já comentando anteriormente. As tecnologias mais comumente utilizadas com esta técnica é o Wi-Fi e o *Bluetooth*. Os *smartphones* "leem" a intensidade dos sinais

transmitidos para realizar o processo do cálculo da localização. Divide-se em duas etapas chamadas de fase *offline* ou de calibração e fase *online* ou de posicionamento.

Durante a fase *offline* o dispositivo móvel captura valores de RSSI e os associa a uma posição do ambiente, que pode ser por exemplo, uma coordenada (e.g., x = 1 e y = 2). Já na fase *online*, etapa em que se deseja obter a localização, são colhidas novas amostras de RSSI e estas são comparadas com as armazenadas na base de dados durante a fase *offline*.A principal vantagem do *fingerprinting* é a sua capacidade em oferecer uma boa precisão. Já sua maior desvantagem é que o processo executado na fase *offline* pode ser trabalhoso e complexo, dependendo do tamanho do ambiente. A utilização desta técnica de localização faz parte da solução proposta neste trabalho, e portanto será abordada em maiores detalhes no capítulo seguinte.

2.5.4 PDR - *Pedestrian Dead Reckoning*

Esta técnica faz uso de sensores tais como acelerômetro, giroscópio e magnetômetro para identificar a movimentação do usuário no ambiente e inferir sua posição. Para isto é necessário conhecer a posição anterior à sua movimentação, e a partir de informações sobre a quantidade de passos dados, e a direção de caminhada é possível detectar a nova posição. Os *smartphones* atuais geralmente são equipados com o chamado *Inertial Measurement Unit* (IMU), que possui estes três sensores embutidos, e isso tem contribuído para estudos que utilizam esta técnica surgirem. Sua maior desvantagem é que como os sensores inerciais não são totalmente precisos, à medida que o usuário se movimenta no ambiente o nível de precisão cai, devido à quantidade de erros de medição que é cumulativo. Para resolver este problema é necessário utilizar alguma outra técnica de localização como uma forma de mitigar estes erros.

A Figura 2.10 apresenta um esquema-resumo de todas as técnicas de localização *indoor* apresentadas neste capítulo.

2.6 Considerações Finais

No decorrer deste capítulo foram apresentados os principais conceitos, métodos e tecnologias que estão relacionados a este trabalho. Entre os temas

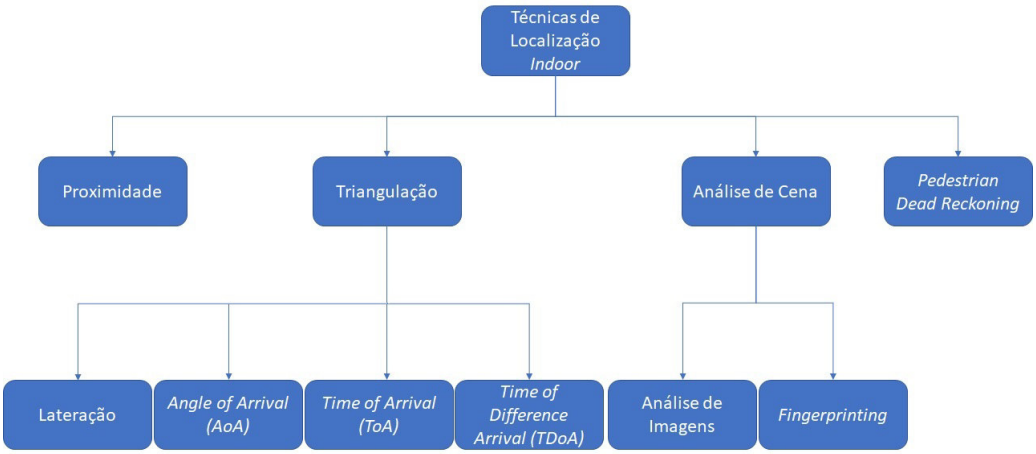

Figura 2.10: Resumo das Técnicas de Localização *Indoor* [44].

abordados estão: Computação Ubíqua e Sensível ao Contexto; Qualidade de Contexto; A Infraestrutura INavigS; Métodos e Tecnologias para Localização *Indoor*.

No que diz respeito à Computação Ubíqua e Sensível ao Contexto, foi apresentado seu conceito e uma breve discussão sobre as diferentes classificações para contexto, definidas por diferentes autores. Além disso, foi abordado um modelo arquitetural conceitual, considerado bem aceito na literatura e utilizado com o intuito de guiar a construção de novas aplicações sensíveis ao contexto sobre como seus componentes devem estar organizados. Por fim, foram citados requisitos importantes dos quais aplicações sensíveis ao contexto precisam ter para desempenharem seu papel corretamente.

Diante das características do cenário em que aplicações da computação ubíqua e sensível ao contexto estão inseridas, um aspecto a ser observado é a chamada Qualidade de Contexto. Isto porque, muitas das fontes de obtenção de dados contextuais (por exemplo, sensores) não são livres de erros, e portanto, não totalmente confiáveis. Isto pode resultar em problemas durante a execução de aplicações sensíveis ao contexto e portanto é necessário observar o nível de qualidade dos dados de contexto consumidos por estas aplicações. Diferentes parâmetros foram encontrados nos diversos estudos da área e os principais foram citados. Pôde-se concluir que não há na literatura uma padronização a respeito da nomenclatura e significado dos parâmetros de QoC, onde cada autor os denomina e expõe seus significados à sua maneira.

Posteriormente, foi mostrada a organização da infraestrutura INavigS, cenário deste trabalho. Dentro da referida infraestrutura foram detalhados cada um de

seus serviços e componentes, além de expor informações sobre o *middleware* M-Hub, que atua no INavigS com o papel de *Context Provider*.

Por fim, foram apresentados os principais conceitos, técnicas e tecnologias utilizadas acerca da área de localização *indoor*. Foi possível identificar que cada uma destas técnicas e tecnologias possuem suas vantagens e desvantagens e que o uso de cada uma delas pode se encaixar de acordo com a necessidade do cenário em que será utilizada. O conhecimento sobre estes conceitos e técnicas serão importantes durante a leitura do Capítulo 4, que trata da proposta deste trabalho.

3 Trabalhos Relacionados

Alguns trabalhos recentes na área da Computação Ubíqua e Ciente de Contexto têm abordado alternativas ao uso do sistema de GPS em ambientes fechados, propondo técnicas para a localização e navegação *indoor*. Neste capítulo iremos apresentar trabalhos relacionados que propõem arquiteturas voltadas para o fornecimento de serviços de navegação em ambientes internos, explorando as características da sua arquitetura, métodos e tecnologias utilizadas.

3.1 InLoc

O InLoC [48] é um sistema de posicionamento e rastreamento para ambientes *indoor* que utiliza *smartphones*. Seu objetivo é oferecer serviços de navegação, fornecendo rotas a partir de um local de origem até o destino desejado. Para calcular a posição são combinadas as técnicas de localização por Trilateração e *Pedestrian Dead Reckoning* (PDR).

Sua arquitetura divide-se em dois componentes. O primeiro é o "*Localization Server*" que tem o papel de calcular a posição do usuário, fornecer mapas do ambiente e processar as instruções de navegação. O outro componente consiste em uma aplicação para dispositivos móveis (e.g., *smartphones*) e opera realizando a leitura de *beacons* BLE próximos além de capturar dados para obter a orientação do usuário.

Para a execução da técnica de PDR é utilizado um filtro de partículas. Entretanto, conforme já exposto na Seção 2.5.4, esta técnica pode acumular erros que aumentam à medida que o usuário se desloca no ambiente, devido à imprecisão dos sensores inerciais.

Por este motivo, a técnica de Trilateração é utilizada: para que sejam corrigidos possíveis erros de localização ocasionados pela técnica de PDR, além de ser necessária para a detecção da posição inicial do usuário. As duas técnicas são combinadas por meio de um estimador *Baysiano*. A orientação do usuário é detectada

a partir da fusão dos dados dos sensores inerciais do smartphone, tais como bússola, giroscópio e acelerômetro.

Beacons BLE são distribuídos no ambiente com uma distância máxima entre eles de até 8 metros. O algoritmo *K-Nearest Neighborg* (KNN) é utilizado para estimar a distância entre o *smartphone* e cada um dos 4 *beacons* BLE com maior intensidade de sinal no momento. O *Least Squares Trilateration* (LST) [49], é aplicado para estimar a localização a partir de valores da distância entre diversos *Point of Interest* (POI). Por fim, para calcular a rota a ser seguida pelo usuário para chegar ao destino pretendido, é utilizado o algoritmo *Dijkstra*.

Uma limitação deste trabalho é que não são oferecidas instruções passo a passo ao usuário durante o percurso, possibilitando apenas que o usuário acompanhe no mapa sua localização atual em relação ao destino pretendido.

3.2 Guide Beacon

O *GuideBeacon* [50] é um sistema de navegação que utiliza *beacons* BLE instalados no ambiente e é executado em *smartphones* convencionais. Entre seus serviços fornecidos estão: navegação, pré-visualização e recalculo da rota por meio de uma narração que descreve todo o percurso, posicionamento e orientação do usuário.

A sua arquitetura divide-se em três componentes: *Beacon Manager*, *Map Database* e o *Guide Beacon App*. Em sua parte física, a arquitetura conta com *beacons* BLE instalados em POI (e.g., portas, salas, escadas, elevadores etc.).

O *Beacon Manager* é responsável por associar os *beacons*, por meio de seu *Universally Unique Identifier* (UUID), aos pontos de interesse. O *Map Database* fornece mapas do ambiente *indoor* para que seja realizado o seu *download* no dispositivo móvel do usuário. O *Guide Beacon App* oferece a interface do usuário e os serviços do *Guide Beacon*, sendo executado em *smartphones*, oferecendo modos de uso por meio de comandos de voz ou textos.

A técnica de localizaçao utilizada é a Proximidade, já apresentada na Seção 2.5.1. Para obter a localização, é calculada a média móvel ponderada de um conjunto de leituras de RSSI associadas a cada um dos *beacons* detectáveis. Para determinar

qual *beacon* está mais próximo, é analisado o conjunto dos valores recebidos, sendo escolhido aquele que tiver a quantidade superior a um limite pré-estabelecido. Para calcular a rota e determinar o menor caminho a ser percorrido, também é utilizado o algoritmo Dijkstra. Informações sobre a orientação do usuário são obtidas a partir da bússola disponível no *smartphone*.

3.3 NavCog

Em Ahmetovic et al. [51] é apresentado o *NavCog*, um sistema de navegação *indoor* voltado para deficientes visuais, de uso também baseado em *smartphones*, que fornece assistência de navegação passo a passo. Além disso, o sistema informa ao usuário sobre pontos de interesse próximos e a presença ou não de acessibilidade durante a rota.

A arquitetura do NavCog é composta por três componentes. O *Map Server* armazena e fornece informações sobre o ambiente, como a presença de escadas, elevadores, rampas de acesso etc. Em sua camada física existem os *beacons* BLE, utilizados para obter a localização. O *NavCog App* é uma aplicação móvel que guia o usuário a um destino pretendido a partir das informações obtidas sobre a localização do usuáiro e do mapa do ambiente. A Figura 3.1 apresenta a arquitetura do NavCog e seus componentes.

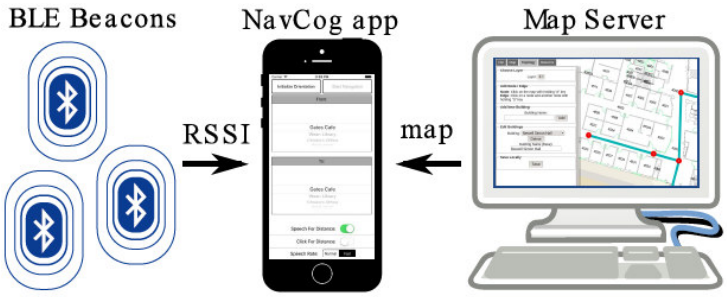

Figura 3.1: Arquitetura do NavCog [51].

Neste trabalho foi aplicada a técnica de localização *fingerprinting*, utilizando o algoritmo KNN para estimar a localização a partir da comparação dos valores obtidos com leituras do RSSI. Para calcular a orientação do usuário, é utilizado o sensor de giroscópio.

Além disso, uma aplicação Web foi desenvolvida para facilitar a construção dos mapas e a adaptação do sistema em diferentes ambientes. Durante a sua execução existem três opções de uso durante a assistência de navegação: anúncios sobre distância através de comandos de voz sintetizada ou alertas sonoros; instruções de ações durante o percurso como, por exemplo, a mudança de piso no ambiente *indoor*; por fim, tem-se a descrição de POI.

Foram identificadas duas limitações deste trabalho. A primeira, em relação às instruções de navegação, pois a aplicação não é capaz de alertar o usuário sobre desvios da rota. Além disso, conforme mostrado na Figura 3.1, o processamento dos dados de localização e instruções de navegação é feito na aplicação que executa no próprio *smartphone*, o que exige um alto consumo de recursos do sistema e consequentemente de bateria do dispositivo móvel.

3.4 StaNavi

O StaNavi [34] é um sistema de navegação *indoor* desenvolvido no cenário de uma estação de trem, utilizando *smartphones* e *beacons* BLE, que objetiva guiar pessoas com deficiência visual. O projeto deste trabalho se deu na estação de Tókio, uma das mais movimentadas do mundo. Os serviços fornecidos são: localização do usuário, informações sobre pontos de interesse, navegação, interface otimizada para atender deficientes visuais e avisos sobre desvio da rota.

A arquitetura do StaNavi é composta por três componentes. Na parte física da arquitetura, são encontrados diversos *beacons* BLE, espalhados pelo ambiente. Existe também uma aplicação para *smartphones* e uma aplicação que roda em um servidor que fornece as instruções de navegação da origem ao destino. A técnica de localização utilizada é a proximidade e a orientação do usuário é obtida por meio da bússola do *smartphone*.

Os *beacons* BLE foram distribuídos próximos a pontos de referência tais como elevadores, banheiros, salas etc. Quando o *smartphone* detecta o UUID de um *beacon* pré-definido, o aplicativo do StaNavi detecta que o usuário está próximo a um dos pontos de referência e oferece as instruções necessárias para o usuário chegar do ponto atual até o ponto de referência pretendido.

3.5 *Building a Practical Wi-Fi-Based Indoor Navigation System*

No trabalho de Han et al. [52] é abordado o processo de desenvolvimento de um sistema de navegação *indoor* utilizando a tecnologia de transmissão Wi-Fi e *smartphones*. A metodologia de localização utilizada foi o *fingerprinting*. O sistema de navegação apresentado neste trabalho foi implantado no Instituto Coreano de Ciência e Tecnologia.

Por utilizar a tecnologia Wi-Fi, o primeiro passo foi analisar os *access point* disponíveis no ambiente para, caso necessário, realizar a instalação de novos dispositivos, se a quantidade for insuficiente. Devido ao ambiente ser muito grande, percebeu-se que houve muitos locais com sinais de Wi-Fi fracos ou inexistentes, então foi necessário instalar cerca de 200 novos APs.

O próximo passo foi desenhar o mapa do ambiente *indoor*, que tem como finalidade direcionar os usuários do sistema e marcar os locais, através de coordenadas com eixo x e y, os quais seriam realizadas as leituras dos valores de RSSI. Depois de marcar os pontos de coleta no mapa, o próximo passo foi realizá-la. Foi coletada uma quantidade de 20 a 30 amostras para cada ponto.

Após armazenar as amostras em uma base de dados, é executado o mecanismo de localização, que roda o algoritmo *Weighted K-Nearest Neighbor* (WKNN) e um filtro de *Kalman* adaptativo, uma variante do filtro de *Kalman* estendido. Um módulo de calculo de rota encontra o caminho ideal entre uma localização atual do usuário até o destino pretendido. Entre os destinos estão POI registrados, tais como lojas, escritórios etc. A definição dos caminhos a serem percorridos pela rota é feita utilizando o algoritmo Dijkstra.

3.6 Discussão

A Tabela 3.1 compara os sistemas de navegação *indoor* apresentados anteriormente. Os critérios para comparação foram:

Tabela 3.1: Tabela comparativa entre os trabalhos relacionados.

	Técnica de Localização	**Tecnologia de Transmissão Sem-Fio**	**Cenário Abordado**	**Processamento**
Chandel et al. [48]	Trilateração	BLE	Genérico	Servidor
Cheraghi et al. [50]	Proximidade	BLE	Genérico	Servidor
Ahmetovic et al. [51]	Fingerprinting	BLE	Genérico	Smartphone
Kim et al. [34]	Proximidade	BLE	Específico	Servidor
Han et al. [52]	Fingerprinting	Wi-Fi	Específico	Servidor

- **Método de localização utilizado**: analisa o método de localização utilizado, dentre os já apresentados na Seção 2.5;
- **Tecnologia de transmissão sem-fio utilizada**: verifica a tecnologia utilizada na infraestrutura física da arquitetura para a obtenção da localização;
- **Cenário da aplicação**: analisa se o trabalho aborda uma solução para um cenário representado por um local específico ou se foi feita uma abordagem genérica, capaz de ser implementada em qualquer tipo de cenário *indoor*;
- **Processamento dos dados**: verifica se o calculo da localização é feito no próprio dispositivo móvel ou se existe um módulo do lado do servidor que desempenha este papel;

3.6.1 Método de Localização utilizado

A comparação mostra que os trabalhos utilizam diferentes métodos de localização. Em Chandel et al. [48] é possível encontrar uma aplicação de navegação *indoor* baseado na técnica de Trilateração, que utiliza o algoritmo KNN para estimar a distância entre o dispositivo móvel e os *beacons*, e o método LST para calcular a localização.

Já o trabalho de Cheraghi et al. [50] utiliza a técnica de localização por Proximidade em que a idéia é calcular a média ponderada móvel sobre um conjunto de amostras de RSSI associadas a cada *beacon* e avaliar se esse valor é superior a um certo limiar. Esta operação é realizada apenas para os *beacons* que foram detectados por uma quantidade mínima de vezes dentro de um intervalo de tempo.

A técnica de localização por proximidade também foi utilizada em Kim et al. [34]. Neste trabalho a busca por *beacons* ocorre a cada dois segundos. Os *beacons* foram inseridos próximos a POI do ambiente interno e a técnica utilizada detecta a proximidade do usuário com estes pontos e traça rotas a partir desta informação. Tanto Ahmetovic et al. [51] como Han et al. [52] utilizaram a técnica de *fingerprinting*. Para isso, o primeiro implementou o algoritmo KNN e o segundo utilizou o algoritmo WKNN.

3.6.2 Técnica de transmissão sem-fio utilizada

Conforme mostrado na Seção 2.5 as técnicas de localização *indoor* podem ser combinadas com tecnologias de transmissão sem-fio, como Wi-Fi, BLE etc. Entre os trabalhos apresentados, Chandel et al. [48], Cheraghi et al. [50], Ahmetovic et al. [51] e Kim et al. [34] utilizaram BLE. Já Han et al. [52] em sua arquitetura proposta, utilizou Wi-Fi.

Cada uma destas tecnologias apresentam vantagens e desvantagens. Em relação ao Wi-Fi, a principal vantagem é a sua presença em infraestruturas de redes sem fio existente na maioria dos ambientes *indoor*. Como desvantagem, o Wi-Fi pode atuar na banda 2.4 ou 5 Ghz, necessitando que o dispositivo móvel realize o escaneamento nestas duas frequências. Isto aumenta o tempo de varredura para detectar o SSID e o RSSI de cada ponto de acesso, podendo demorar vários segundos, ocasionando atrasos na atualização do posicionamento.

Algumas plataformas móveis permitem acesso a parâmetros de varredura apenas das redes Wi-Fi em que estão conectados, como os *smartphones* da marca *Apple*, que utilizam o sistema operacional iOS. Arquiteturas baseadas em Wi-Fi ficam limitadas à utilização de dispositivos móveis que executem sistemas operacionais capazes de realizar leituras de Wi-Fi de todas as redes sem fio presentes no ambiente, mesmo que não estejam conectados a elas, como o *Android*. Uma outra desvantagem é o consumo de bateria, que é maior quando se utiliza Wi-Fi em relação ao BLE.

A principal desvantagem do BLE é a necessidade de instalar uma infraestrutura física, composta de *beacons*, o que aumenta o custo de implementação de arquiteturas que venham a utilizar esta tecnologia. Apesar disto, seu uso tem

se mostrado mais vantajoso que o Wi-Fi em termos de desempenho e precisão de localização.

3.6.3 Cenário da aplicação

Nos trabalhos de Chandel et al. [48], Cheraghi et al. [50] e Ahmetovic et al. [51], foram desenvolvidas arquiteturas que podem ser utilizadas de maneira genérica, ou seja, com a possibilidade de serem estendidas a outros cenários, sendo soluções reutilizáveis. Entretanto, em Kim et al. [34] e Han et al. [52] são apresentadas arquiteturas focadas em oferecer serviços de navegação em um cenário específico.

O trabalho de Kim et al. [34] aborda a experiência de implementação de um sistema de navegação *indoor* em uma grande estação de trem na cidade de Tóquio, no Japão, uma das mais movimentadas do mundo, com uma movimentação de mais de 400.000 passageiros e 3.700 trens por dia. Um ambiente tão grande e movimentado resultou em vários problemas práticos como erros de localização e nos dados sobre orientação devido a falhas na leituras do sensor de bússola do *smartphone*, causadas pela presença de altos níveis de ruídos e uma grande quantidade de obstáculos.

Em Han et al. [52] é descrita a implementação de uma arquitetura que fornece serviços de navegação para o Complexo COEX em Seul na Coreia do Sul, que compreende a maior área de shopping center subterrâneo da Ásia, com três hotéis, uma torre de escritórios de 55 andares e uma outra com 41 andares, além de uma estação de metrô, e um terminal de aeroporto, com uma área de mais de 450.000 m^2.

Apesar de eficazes, os trabalhos de Kim et al. [34] e Han et al. [52] são muito específicos para o cenário que abordam e não oferecem a possibilidade de extensão para serem implantados em outros ambientes.

3.6.4 Processamento dos dados

As técnicas de localização abordadas em todos os trabalhos apresentados necessitam de processamento através de algoritmos que transformam dados brutos, por exemplo, os valores de RSSI, em dados de contexto sobre a localização.

Este aspecto foi analisado em cada um dos trabalhos apresentados e foi identificado que as arquiteturas propostas em Chandel et al. [48], Cheraghi et al. [50], Kim et al. [34] e Han et al. [52] possuem componentes que executam externamente ao *smartphone* e realizam o processamento dos dados de contexto sobre localização, utilizando uma abordagem *server-side*. Já em Ahmetovic et al. [51] o cálculo da localização é feito no próprio dispositivo móvel.

Realizar o cálculo da localização no próprio dispositivo móvel pode apresentar como vantagem a independência em relação à rede, já que não há a necessidade de troca de informação constante entre o *smartphone* e componentes externos a ele, logo mesmo que haja queda de conexão o sistema continuará funcionando normalmente. Entretanto, atribuir toda a carga de processamento ao dispositivo móvel pode trazer desvantagens como o aumento do consumo de bateria, além de comprometer o desempenho da aplicação, já que o poder computacional de um dispositivo móvel pode ser inferior ao de um servidor dedicado, por exemplo.

3.7 Considerações Finais

No decorrer deste capítulo foram exploradas arquiteturas voltadas para fornecer serviços de navegação *indoor* que utilizam dispositivos móveis, como os *smartphones*, e tecnologias de comunicação sem fio. Foram analisados os trabalhos de Chandel et al. [48], Cheraghi et al. [50], Ahmetovic et al. [51], Kim et al. [34] e Han et al. [52], expondo suas principais características. Foram analisados detalhes como a técnica de localização abordada, tecnologia de transmissão sem-fio utilizada para obtenção da localização, o cenário em que foi implementada, se para uso genérico ou para um local específico e o lado do processamento dos dados de contexto sobre localização, se no próprio dispositivo móvel ou no servidor.

4 Solução Proposta

Este capítulo irá abordar os problemas associados a Qualidade de Contexto identificados na infraestrutura INavigS. Entre os problemas estão: a baixa precisão nos dados de contexto referentes à localização, ausência de dados de contexto sobre a orientação do usuário e o alto tempo de resposta entre o período em que o *smartphone* do usuário captura pacotes originados pelos *beacons* BLE e a exibição das instruções de navegação. Em seguida, serão apresentados os parâmetros de Qualidade de Contexto aplicados para resolver estes problemas (*precision*, *completeness* e *up-to-dateness*). Por fim, serão mostrados os métodos e técnicas utilizados e adaptados para abordar a problemática em questão, que são: a localização por *fingerprinting*, para melhorar a precisão da localização; o uso dos sensores inerciais do *smartphone* para calcular a orientação do usuário; o cálculo do parâmetro *up-to-dateness* para identificar a idade dos dados de contexto e descartar informações consideradas antigas ou inválidas.

4.1 Problemas Identificados na Infraestrutura INavigS

O INavigS é uma infraestrutura de software sensível ao contexto que objetiva oferecer serviços de navegação *indoor*, isto é, em ambientes fechados, onde os sensores que captam sinais de GPS não funcionam adequadamente.

Apesar de prover os serviços aos quais a infraestrutura se propõe, foram identificados alguns problemas relativos a:

- Baixa precisão dos dados de contexto sobre a localização do usuário devido ao método de localização utilizado, baseado em proximidade;
- Dados de contexto incompletos ou inexistentes sobre a orientação do usuário, pois a infraestrutura obtém a orientação a partir do valor de azimute do último trecho percorrido;

- Alto tempo de resposta entre a obtenção da localização do usuário e a exibição de instruções de navegação devido a possibilidade de conectividade fraca ou intermitente de redes sem-fio;

Baixa precisão dos dados de contexto sobre a localização do usuário

Conforme mostrado no Capítulo 2, a infraestrutura INavigS baseia-se no método de localização *indoor* denominado Proximidade, já apresentado na Seção 2.5.1.

A técnica, que consiste em determinar a localização do usuário de acordo com a distância entre o seu dispositivo móvel e pontos de referência, tem como vantagem a sua facilidade de implementação.

Na infraestrutura, os pontos de referência são pequenos dispositivos, denominados *beacons*, que transmitem sinais via tecnologia BLE. A partir da intensidade dos sinais capturados pelo dispositivo móvel, é possível calcular, de maneira aproximada, a distância entre o usuário e um *beacon*. A área de abrangência do *beacon* mais próximo definirá a localização do usuário.

Esta técnica oferece como principal desvantagem a sua baixa precisão, que depende da quantidade de pontos de referência, isto é, o número de *beacons* BLE utilizados. Além da baixa precisão, com o seu uso não é possível determinar a posição absoluta do usuário dentro de uma área. Isto significa que é possível detectar apenas que o usuário está localizado ou próximo de determinada sala ou ambiente do prédio, entretanto não é possível identificar exatamente em qual local ele está.

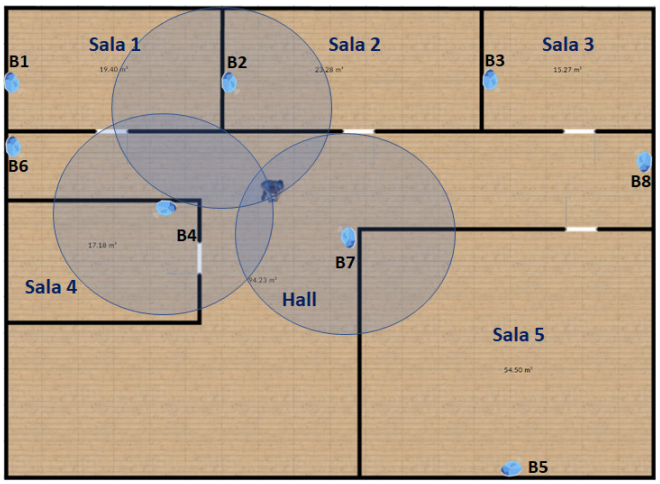

Figura 4.1: Exemplo de funcionamento da técnica de Localização por Proximidade.

A Figura 4.1 exemplifica o funcionamento da técnica de localização por proximidade utilizada atualmente pela infraestrutura INavigS. O smartphone do usuário detectou a presença de três *beacons* próximos: "B2","B4" e "B7". Logo, o usuário pode estar localizado na Sala 2, Sala 4 ou no *Hall* do prédio. Para determinar a localização do usuário, a intensidade de sinal (RSSI) de cada um dos *beacons* é comparada.

Tabela 4.1: Valores de RSSI dos *beacons* mostrados na Figura 4.1

Beacon	Localização	RSSI
1	Sala 1	-
2	Sala 2	-72 dBm
3	Sala 3	-
4	Sala 4	-68 dBm
5	Sala 5	-
6	Hall	-
7	Hall	-47 dBm
8	Hall	-

A Tabela 4.1 mostra uma tabela com os valores de RSSI dos *beacons* detectados. Entre eles, aquele que apresentou o maior nível de RSSI foi o *beacon* "B7". Logo, é possível inferir que o usuário está mais próximo dele, e portanto localizado no Hall do prédio ou dentro da área que corresponde a sua abrangência, representada pelo círculo azul na Figura 4.1.

Para aplicações de navegação em ambientes internos, o parâmetro precisão é fundamental para o fornecimento dos serviços de maneira satisfatória. Por exemplo, na Figura 4.1 se a precisão de localização for corresponde à área de abrangência do *beacon* "B7", não será possível definir se o usuário está localizado no Hall do prédio ou na Sala 5.

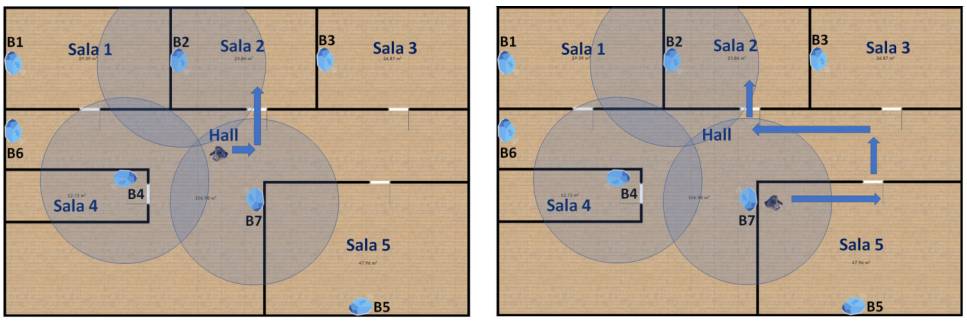

Figura 4.2: As instruções de navegação devem mudar de acordo com a localização do usuário.

A Figura 4.2 mostra que a instruções de navegação dependem diretamente da informação sobre a localização do usuário. No exemplo mostrado na figura, a

infraestrutura INavigS não é capaz de identificar, pelo nível de precisão conseguido, se o usuário está realmente na Sala 5 ou no *Hall* do prédio, mas apenas que ele está mais próximo do *beacon* B7. Dependendo de qual dos dois lugares o usuário estiver localizado, instruções de navegação diferentes deverão ser apresentadas, caso ele necessite, por exemplo, chegar na Sala 2. Este problema pode gerar um grande número de erros durante o cálculo das instruções de navegação.

Ausência de dados de contexto sobre a orientação do usuário

Durante a execução de aplicações baseadas na infraestrutura INavigS são calculadas as instruções de navegação sempre que a localização do usuário muda. A partir desse momento, são apresentas informações como “vire à direita”, “vire à esquerda", “siga em frente” etc., como mostra a Figura 4.3. Para isto, é preciso conhecer a orientação do usuário, que é dada pelo cálculo do azimute – ângulo que indica uma direção em relação ao norte magnético da terra.

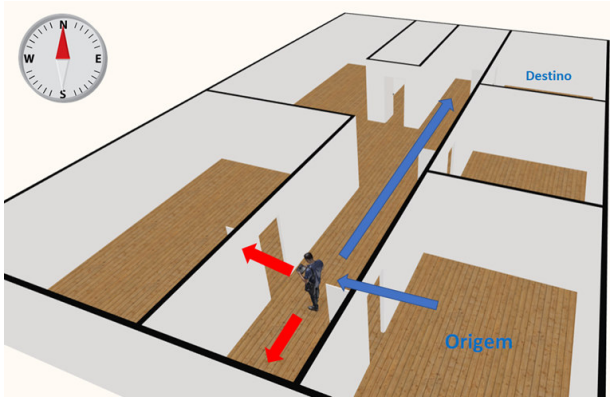

Figura 4.3: Instruções de navegação com orientação.

A Figura 4.4 mostra um exemplo de como o valor do azimute representa uma direção em relação ao norte. Se o valor do ângulo azimute medido for igual a 0º, significa que o usuário está indo em direção ao norte; se o valor for igual a 90º, então o usuário está indo em direção ao leste, e assim por diante.

O INavgiS pressupõe que a orientação do usuário é igual ao azimute do trecho atualmente percorrido. A Figura 4.4 mostra um exemplo de como o cálculo é feito. Assim que a infraestrutura detecta que o usuário saiu da posição 1 e seguiu rumo à posição 2, com base em informações sobre as coordenadas geográficas destas duas posições previamente armazenadas em uma base dados, é calculado o ângulo do

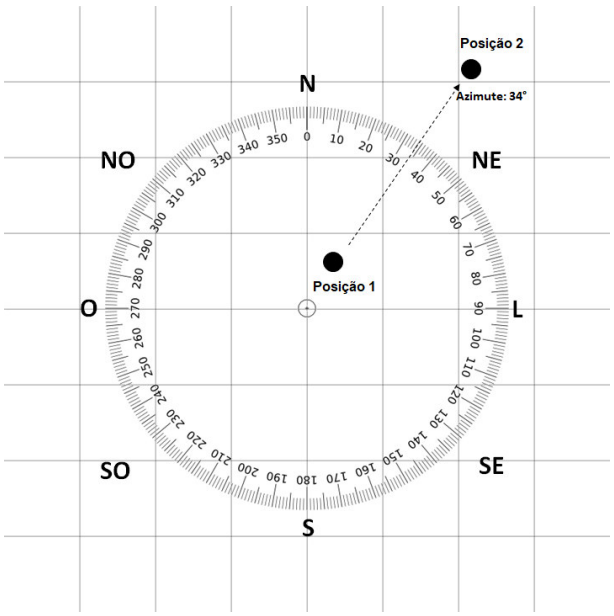

Figura 4.4: Valor de azimute correspondentes às direções.

trecho percorrido. Considerando-se que o valor do ângulo é 45º, significa que o usuário movimentou-se no sentido Nordeste. O cálculo do azimute depende totalmente de informações sobre a localização do usuário.

No início do trajeto, quando ainda não houve mudança de posição, o INavigS considera sempre que o usuário está voltado para o primeiro trecho. Caso isto não seja verdade, a instrução de navegação fornecida estará errada, logo isto pode ser entendido como uma limitação.

Alto tempo de resposta entre a inferência dos dados de contexto sobre a localização e apresentação das instruções de navegação

Conforme mostrado na subseção 2.4 a arquitetura do INavigS é composta por diversos componentes divididos em camadas, que são distribuídos e executam diferentes serviços que exigem uma troca constante de informações. O Provedor de Contexto envia dados brutos sobre os *beacons* detectados para o *Context Manager* que armazena e os fornece para o *Context Reasoner*, que por sua vez, realiza a inferência dos dados a fim de obter as informações de contexto sobre a localização do usuário e fornece as instruções de navegação para a camada superior, *Application Layer*.

Seguir todas estas etapas citadas acima pode resultar em um alto tempo de resposta entre a captura dos dados brutos até que seja obtida a localização e o calculo das instruções de navegação. Além disso, o cenário de funcionamento do INavigS

envolve a comunicação por meio de redes sem fio, que é propenso a conectividade fraca ou intermitente. Isto resulta em falhas que podem causar atrasos na entrega dos dados para os diferentes componentes, tornando as informações de contexto e as instruções de navegação desatualizadas.

Atrasos no funcionamento da infraestrutura podem resultar em instruções de navegação desatualizadas, que podem levar a desvios da rota ou fazer com que o usuário não consiga chegar ao local pretendido.

4.1.1 Aplicação de Parâmetros de QoC na Infraestrtutura INavigS

Nesta subseção serão discutidos resumidamente, os parâmetros de QoC abordados neste trabalho para solucionar os problemas citados acima.

Precision

O parâmetro *Precision* é definido por Neisse et al. [53] como o quão próximo um valor medido por um sensor, que venha a ser utilizado como dado de contexto, está do valor real. Para Buchholz et al. [3], o parâmetro descreve como exatamente uma informação de contexto reflete a realidade.

Dependendo do autor o parâmetro é também conhecido por *Accuracy*, como pode ser visto em Kim et al. [33], que define o parâmetro como o grau em que uma informação de contexto é precisa e confiável.

Assim, é importante observar que os conceitos de precisão e acurácia podem ser considerados distintos. Medidas com acurácia são aquelas em que o valor médio se aproxima do valor correto. Já medidas consideradas precisas são aquelas que apresentam uma pequena dispersão.

Completeness

Este parâmetro indica se uma informação de contexto está completa, disponível, suficiente e não ausente [32] [33]. Segundo Manzoor et al. [30] este parâmetro representa uma relação entre a quantidade de informações de contexto esperadas por uma aplicação ciente de contexto com o número de valores recebidos.

Considerando que a infraestrutura INavigS não conta com dados de contexto sobre a orientação do usuário obtidas por meio de sensores, consideramos que em algumas situações as informações de contexto podem ser incompletas, como já citado na Subseção 4.1.

Up-to-dateness

De acordo com Buchholz et al. [3], este parâmetro representa a idade de uma informação de contexto. Em Manzoor et al. [30] é dito que o parâmetro *Up-to-dateness* é o grau de validade de uma informação de contexto por período de tempo, definido pelo tempo transcorrido entre o instante de medição da informação e o instante atual. É conhecido como *Age* em Manzoor et al. [30] e *Freshness* em Kim et al. [33] e em Sheikh et al. [31], que afirma que este parâmetro indica o período de tempo decorrente entre a determinação da informação até a sua entrega ao consumidor de contexto. Para Huebscher et al. [37] o parâmetro *Up-to-dateness* descreve a frequência com que é possível ou desejado receber novos valores de contexto.

Para Nazario et al. [32] esta variável pode ser chamada de "tempo de vida" e pode ser definida por um valor em que a informação torna-se "velha" ou desatualizada. É considerado como um valor lógico que expressa se a infraestrutura deve garantir a recuperação da versão mais recente dos dados em Corradi et al. [54].

Este é um parâmetro bastante dinâmico que deve ser calculado sempre que a aplicação em questão necessitar do dado de contexto para a tomada de alguma decisão. De acordo com Buchholz et al. [3] o parâmetro pode ser especificado a partir da adição de um *timestamp* às informações de contexto.

4.1.2 Aplicando o parâmetro Precisão (*Precision*) nas informações de contexto sobre localização

Para abordar o problema relacionado à precisão nos dados de contexto referentes à localização do usuário, foi utilizada a técnica de localização por análise de cena, mostrada resumidamente na Subseção 2.5.3.

A técnica consiste em analisar um conjunto de características do ambiente que são capazes de definir uma localização. Estas características podem ser extraídas

por meio da análise de imagens ou através da leitura dos sinais eletromagnéticos presentes no ambiente, neste caso sendo denominada de *fingerprinting*.

Os sinais eletromagnéticos analisados são transmitidos por dispositivos de comunicação sem fio, que utilizam tecnologias tais como Wi-Fi, ZigBee, *Bluetooth* etc. Em Orujov et al. [55], foi realizada uma comparação de desempenho, em termos de precisão, sobre as principais técnicas de localização encontradas na literatura. A conclusão do estudo foi que entre as técnicas analisadas, a que se mostrou mais precisa foi o *fingerprinting*.

Na prática, a metodologia do *fingerprinting* consiste em comparar os valores lidos em tempo real com um conjunto de outros valores previamente armazenados em uma base de dados, chamada de *Fingerprint Map*. Tipicamente, a propriedade analisada é o valor do RSSI. Outras também podem ser utilizadas, como o *Signal to Noise Ratio* (SNR).

Neste trabalho, nos baseamos no valor de RSSI, que tem se mostrado mais eficiente que o SNR, que sofre de flutuações e ruídos que resultam em valores aleatórios e imprevisíveis [7]. O uso do RSSI parte do pressuposto que à medida em que se distancia do transmissor, a intensidade do sinal diminui, e à medida em que se aproxima seu valor aumenta, tornando possível sua associação a localizações.

O *fingerprinting* divide-se em duas fases: a primeira chamada de fase de calibração, ou fase *off-line*, e a segunda denominada fase *on-line* ou fase de posicionamento.

Fase *Off-line* ou Fase de Calibração

Esta etapa consiste em mapear e dividir o ambiente em subáreas, ou posições, como mostrado na Figura 4.5. Cada posição é representada como uma coordenada, com eixos x e y.

A Figura 4.5 ilustra o ambiente mapeado. Cada um dos pontos em vermelho representam uma coordenada ou posição à qual o INavigS passa a identificar. Existem 3 *beacons* BLE espalhados no ambiente. Para cada uma das posições são capturadas amostras de RSSI dos sinais emitidos pelos *beacons*. A leitura dos sinais é feita por um *smartphone* convencional compatível com a tecnologia BLE.

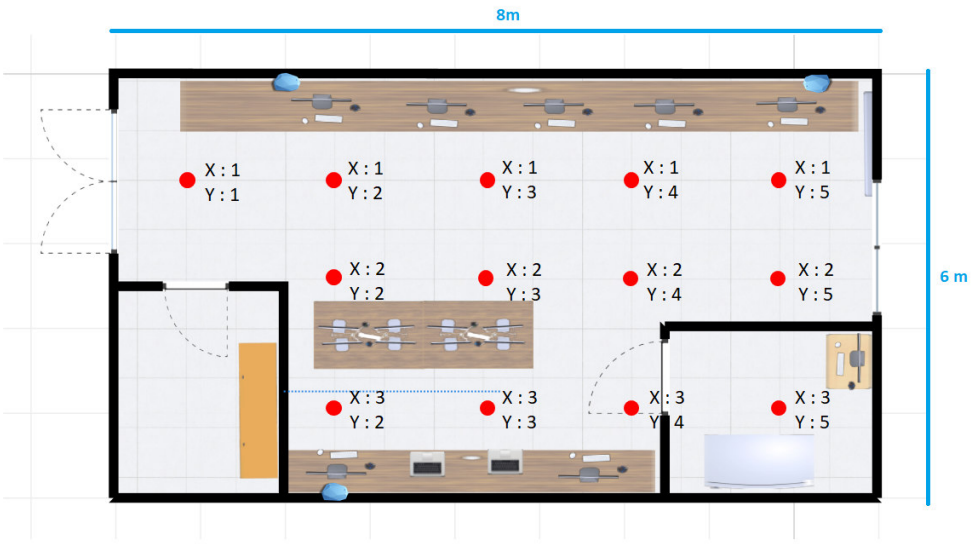

Figura 4.5: Ambiente dividido em posições.

O objetivo desta fase é salvar os valores de RSSI no *Fingerprint Map*, associando-os à posição em que foram lidos. Para isto, o *smartphone* deve permanecer estático enquanto faz a leitura dos sinais. O procedimento se repete até que uma quantidade pré-definida de amostras tenha sido alcançada.

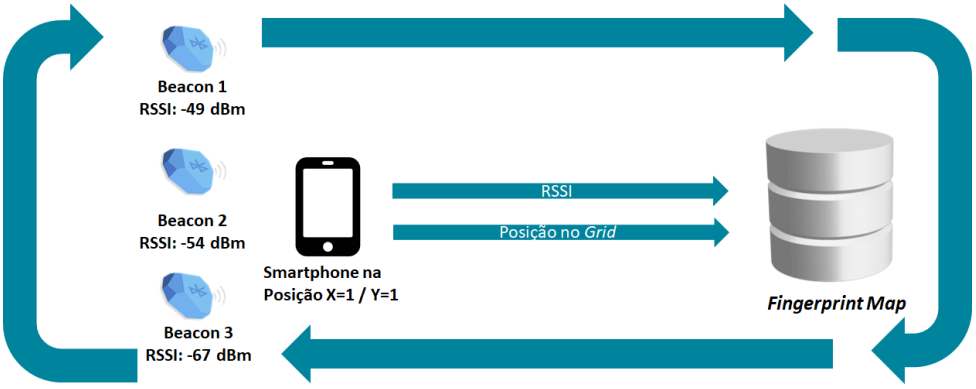

Figura 4.6: O processo de leitura dos valores de RSSI se repete até que a quantidade pré-definida de amostras tenha sido alcançada.

A Figura 4.6 ilustra a etapa. O *smartphone* permanece na posição representada pela coordenada x=1, y=1, lendo os valores RSSI, associando-os à posição em que se encontra e salvando-os no *Fingerprint Map*. O processo é feito para cada uma das posições mostradas na Figura 4.5.

Terminado o processo, o *Fingerprint Map* deverá armazenar as amostras correspondentes a cada um dos três *beacons* para cada uma das posições. A Figura 4.7 representa a estrutura do *Fingerprint Map* após a execução da fase *off-line*.

Número da Amostra	RSSI do Beacon 1	RSSI do Beacon 2	RSSI do Beacon 3	Coordenada
1	- 49 dBm	- 54 dBm	- 67 dBm	X = 1 / Y = 1
2	- 47 dBm	- 50 dBm	- 62 dBm	X = 1 / Y = 1
3	- 47 dBm	- 51 dBm	- 65 dBm	X = 1 / Y = 1
4	- 42 dBm	- 56 dBm	- 59 dBm	X = 1 / Y = 2
5	- 55 dBm	- 47 dBm	- 70 dBm	X = 1 / Y = 2
6	- 53 dBm	- 48 dBm	- 72 dBm	X = 1 / Y = 2
...	...	...	...	...
200	- 40 dBm	-54 dBm	- 70 dBm	X = 3 / Y = 5

Figura 4.7: Representação do *Fingerprint Map*.

Os *beacons* são pequenos dispositivos alimentados por bateria que operam transmitindo pacotes denominados *advertising packets* por meio da tecnologia BLE. Cada *advertising packet* contém um valor de RSSI e um UUID, composto por 128 bits, que identifica individualmente o *beacon* que originou o pacote.

A Figura 4.8 mostra a interface gráfica de uma aplicação *Android*, desenvolvida no escopo deste trabalho, com o objetivo de executar a fase *off-line*.

Antes de iniciar o processo, deve ser preenchido o campo "Localização Atual" que irá representar a posição em que o *smartphone* está posicionado. Após isto, a quantidade de amostras a serem capturadas para cada *beacon*. O campo "Amostras Capturadas" indica a quantidade de amostras que já foram lidas. Em "Amostras Coletadas" é possível observar os valores de RSSI lidos. Assim que o usuário pressiona o botão "Iniciar Scan"o processo já mostrado na Figura 4.6 se repete até que a quantidade de amostras definida seja obtida.

O Algoritmo 1 detalha o procedimento executado na aplicação *Android* durante a fase *offline* para gerar o *Fingerprint Map*. Inicialmente deve ser informada a coordenada correspondente à posição a qual as amostras serão capturadas e a quantidade de amostras que deverão ser armazenadas para cada *beacon*. O processo de escaneamento dos *advertising packets* através da interface BLE do dispositivo móvel se inicia e se repete até que a quantidade de amostras desejadas para cada beacon seja armazenada.

Assim que um *advertising packet* é recebido, o os valores do UUID e do RSSI são extraídos. O UUID é verificado e caso pertença a um dos *beacons* pré-definidos, o valor do RSSI é inserido em uma lista.

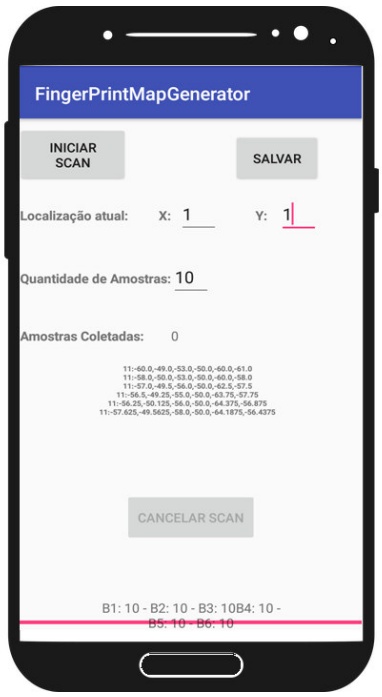

Figura 4.8: Aplicativo *Android* para capturar amostras durante a fase *off-line*.

Quando o número de amostras necessárias é atingido, é realizado o cálculo da média dos valores de RSSI obtidos. Este procedimento é feito para mitigar variações nos valores de RSSI causados, por exemplo, por refrações e efeito multi-percurso. Por fim, os valores são salvos no *Fingerprint Map*.

Algoritmo 1 GERAR O *Fingerprint Map*

Entrada:
Define a coordenada correspondente à posição atual
Define a quantidade de amostras a serem capturadas
Saída: Fingerprint Map

```
1  início
2      enquanto amostrasCapturadas < qtdeAmostrasParaCapturar faça
3          Escaneia advertising packets BLE
4          Extrai o UUID e RSSI
5          Verifica se o advertising packet pertence a um beacon pré-definido
6          se pertence a um beacon pré-definido então
7              Armazena amostra amostras em uma lista
8          fim
9      fim
10     Calcula média
11     Salvar Fingerprint Map
12 fim
```

Após a construção do Fingerprint Map, a fase *online* estará pronta para ser executada.

Fase *Online* ou Fase de Posicionamento

A fase *online*, também chamada de fase de posicionamento, é a etapa em que se irá, de fato, calcular a localização do usuário. Ela deve ser executada após a construção do *Fingerprint Map*. São realizadas leituras em tempo real dos valores de RSSI dos *beacons* e estes são comparados com as amostras armazenadas anteriormente no *Fingerprint Map*. Para isto são utilizados os chamados *Location Estimation Algorithm* (LEA).

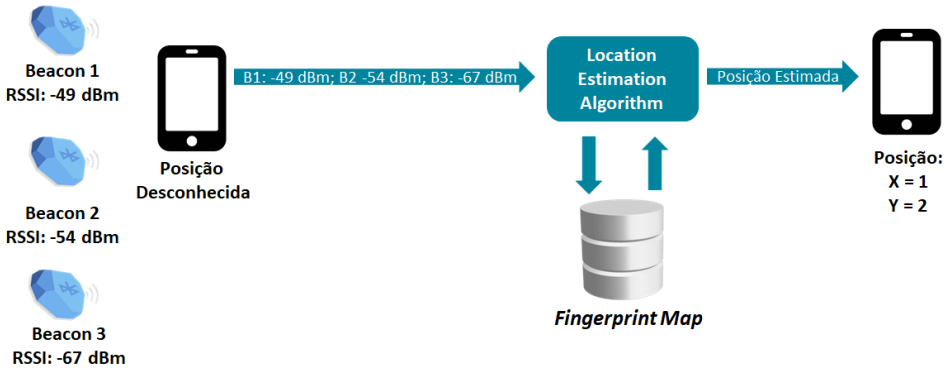

Figura 4.9: Fase *online*.

Os LEA dividem-se em probabilísticos e determinísticos. Na abordagem determinística, as amostras armazenadas no *Fingerprint Map* são comparadas com as obtidas durante a fase *online* utilizando métricas de similaridade, tais como Distância Euclidiana, *Cosine* ou *Tanimoto similarity*. Por outro lado, a abordagem probabilística utiliza métodos estatísticos para calcular a correspondência entre uma amostra obtida na fase online em relação às demais amostras presentes no *Fingerprint Map* utilizando, por exemplo, Teorema de *Bayes*, *Kernel Method*, *Histogram Method*, entre outros.

Uma comparação do desempenho entre as abordagens probabilísticas e determinísticas no cenário de localização *indoor* é feita em Honkavirta et al. [56]. O uso de abordagens determinísticas garantem bons resultados em relação ao nível de precisão. A maior vantagem em se utilizar abordagens determinísticas é a sua simplicidade de implementação e a complexidade computacional relativamente baixa. Além disso, as abordagens probabilísticas requerem muito mais dados de treinamento, tornando a fase *off-line* mais custosa [57].

Neste trabalho utilizamos uma abordagem determinística baseada no cálculo da Distância Euclidiana, que é definida pela equação 4.1, em que D indica o valor da Distância Euclidiana, n a quantidade de amostras armazenadas no *Fingerprint*

Map, *a* representa o valor de RSSI obtido na fase *online*, e *b* o conjunto de valores presentes no *Fingerprint Map*.

$$D = \sqrt{\sum_{i=1}^{n}(a_i - b_i)^2} \tag{4.1}$$

Para isto, foi implementado um módulo que executa o KNN, um algoritmo de aprendizagem de máquina supervisionado que objetiva localizar as *k* amostras mais próximas em relação a uma amostra ainda não classificada. O algoritmo se mostrou satisfatório para lidar com a classificação de dados muito ruidosos, que é o caso do RSSI [58].

Utilizando o KNN, a amostra obtida na fase *online* é comparada com todas as demais previamente armazenadas no *Fingerprint Map*, seguindo o cálculo mostrado na Equação 4.1. O rótulo da amostra que obtiver a menor distância Euclidiana irá definir a localização do usuário. O rótulo pode ser entendido como o valor presente na coluna 5 da tabela mostrada na Figura 4.7, que mostra o valor das coordenadas correspondentes às posições.

Algoritmo 2 *k Nearest Neighbor*

Entrada:
Obtem valores de RSSI
Carrega amostras presentes no Fingerprint Map
Define o valor de *k*
Saída: Rótulo de classificação

```
13 início
14     para cada amostra do fingerprint map faça
15         Calcula distância dos valores de RSSI para cada amostra do Fingerprint Map
16         Determina o conjunto das k amostras mais próximas
17         Escolhe o rótulo com mais representantes no conjunto k
18     fim
19 fim
```

O Algoritmo 2 descreve os passos de funcionamento do KNN. Os valores de entrada para o algoritmo são: os valores de RSSI obtidos durante a fase *online*, as amostras armazenadas previamente no *Fingerprint Map* durante a fase offline e o valor de *k*. É calculada a distância Euclidiana dos valores de RSSI para cada amostra presente no *Fingerprint Map*. É obtido um conjunto com as *k* amostras mais próximas. O rótulo com o maior número de amostras representantes no conjunto *k* será o escolhido.

4.1.3 Aplicando o parâmetro *Completeness* adicionando informações de contexto sobre a orientação do usuário

Conforme dito anteriormente na Seção 2.1, aplicações cientes de contexto necessitam de informações sobre o estado atual de uma entidade (pessoa, lugar ou objeto) para executar suas funções. Em aplicações de navegação *indoor*, além da necessidade de informações sobre a localização do usuário, é preciso conhecer a orientação do usuário para que se possa fornecer instruções de navegação corretamente.

Para isto, exploramos o uso de sensores inerciais, presentes na maioria dos *smartphones* modernos. Um de seus propósitos é obter a orientação do usuário apenas utilizando o *smartphone*, sem a necessidade de sensores externos adicionais. Com isto, é possível por exemplo, obter o valor do azimute em tempo real sempre que forem calculadas novas instruções de navegação.

Estes sensores fazem parte do chamado IMU, um pequeno dispositivo eletrônico composto de magnetômetro, acelerômetro e giroscópio. No sistema de navegação *outdoor* baseado em GPS o magnetômetro dos *smartphones* já é largamente utilizado como bússola, entretanto quando se trata de ambientes *indoor* o seu uso é mais limitado, apresentando problemas ocasionados por interferência eletromagnética devido à presença de dispositivos como computadores, ar-condicionado, televisores etc. Portanto, utilizar dados brutos deste sensor pode resultar em erros de medição, ruídos e imprecisão [59].

Para melhorar o uso destes sensores em *smartphones*, são utilizados algoritmos de fusão de sensores, que combinando os dados capturados pelos sensores inerciais, melhoram a qualidade da informação sobre a orientação do usuário, conforme ilustrado na Figura 4.10.

Para obter os dados de orientação do usuário pelos sensores do smartphone, foi utilizada a classe Compass [1] que implementa um algoritmo de fusão que combina os dados obtidos pelo magnetômetro e acelerômetro, resultando no valor de azimute referente à orientação do usuário.

[1] https://github.com/iutinvg/compass

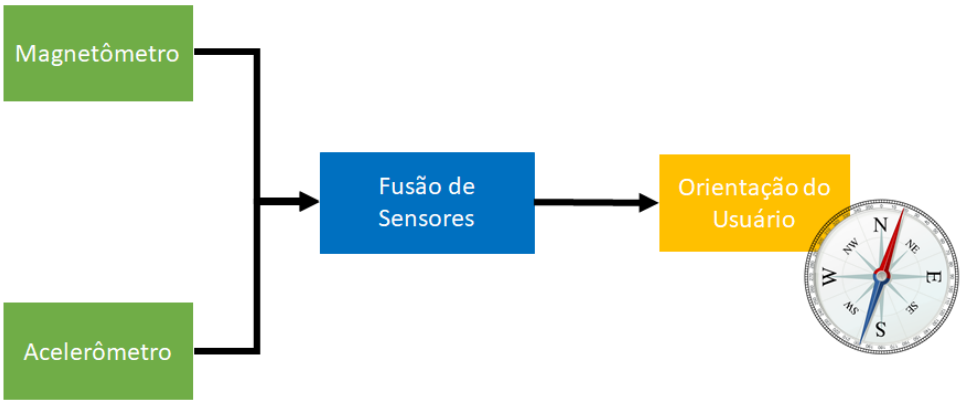

Figura 4.10: Uso de sensores de *smartphone* para obter a orientação do usuário

4.1.4 Aplicando o parâmetro *Up-to-dateness* para tratar do alto tempo de resposta para apresentação de instruções de navegação

As informações de contexto sobre localização são dinânmicas e sofrem mudanças à medida em que o usuário se movimenta através do ambiente. Por esta razão, é essencial observar se uma informação de contexto sobre localização ainda é válida. O parâmetro de QoC associado à validade de uma informação de contexto é denominado como *Up-to-dateness* ou *freshness*.

$$u = 1 - \frac{i}{t_v} \tag{4.2}$$

A equação 4.2, apresentada por Manzoor et al. [60], é utilizada para calcular o grau de confiança, em termos de validade, para se utilizar um dado de contexto em uma aplicação específica em um determinado momento. A variável u representa o valor do parâmetro *Up-to-dateness*, o i indica a idade do dado de contexto e t_v representa o seu tempo de vida.

Para isto é levada em consideração a idade do dado de contexto e o seu tempo de vida. A idade é obtida pela Equação 4.3, onde t_a é o período de tempo atual e t_m o tempo em que o dado foi medido.

$$i = t_a - t_m \tag{4.3}$$

É importante observar que a validade do dado de contexto diminui à medida que sua idade aumenta. Isto significa que um dado de contexto com o parâmetro *Up-to-dateness* baixo indica que este não representa mais o estado atual da entidade a que se refere, neste caso, a localização do usuário já pode ter mudado.

Neste trabalho nós enriquecemos os dados de contexto de localização com o parâmetro *Up-to-dateness*, e fornecemos o valor para a infraestrutura INavigS, de forma que caso seja detectado que o dado possui um valor para o parâmetro *Up-to-dateness* abaixo do esperado, é preferível que este seja descartado e solicitado novamente, em vez de consumido. Isto evita que instruções de navegação utilizem informações muitos antigas, evitando que instruções de navegação erradas sejam exibidas para o usuário além de processamento desnecessário.

4.1.5 Cenário de execução do INavigS com os parâmetros de QoC propostos

Esta subseção descreverá a arquitetura do cenário de execução das soluções propostas acima conforme ilustrado na Figura 4.11. É composta de *beacons* BLE, *smartphone*, *broker* MQTT e um servidor executando os serviços INavigS.

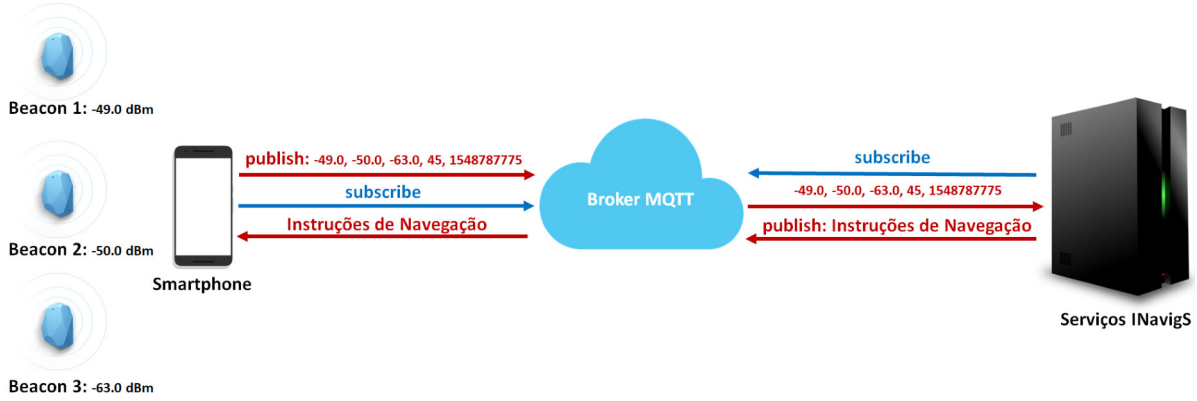

Figura 4.11: Arquitetura Geral.

Beacons BLE

Conforme mostrado na Figura 4.11 os *beacons* permanecem afixados transmitindo *advertising packets*, do qual são aproveitados o valor do UUID, para

identificar o *beacon* que originou o pacote, e parâmetros para calcular a intensidade de sinal (RSSI).

Smartphone

Um *smartphone* realiza leitura dos dispositivos BLE próximos, filtrando os pacotes recebidos pelo UUID, para identificar quais foram originados pelos *beacons*. O objetivo é receber os valores de RSSI para que posteriormente seja calculada a localização do usuário.

Além disso, são utilizados os sensores internos do *smartphone* (acelerômetro e magnetômetro) para calcular o valor do azimute e determinar a orientação do usuário. Os valores de RSSI e do azimute são agrupados junto a um *timestamp* que representa o momento em que esses dados foram coletados e será utilizado para calcular o parâmetro *Up-to-dateness*. A mensagem é então publicada em um tópico, utilizando o protocolo *Message Queue Telemetry Transport* (MQTT).

Broker MQTT

O protocolo MQTT é voltado para a troca de mensagens em redes de sensores sem fio. É utilizado neste trabalho devido a sua capacidade em lidar com problemas e características típicas de cenários que envolvem o uso de sensores, dispositivos móveis e redes sem fio, tais como:

- Perda temporária de *link* de conexão;
- Falhas e substituição de nós da rede;
- Heterogeneidade de tecnologias de comunicação em redes e a presença de diferentes esquemas de endereçamento;
- Ambientes de redes com alta latência e baixa largura de banda
- Utilização de dispositivos com recursos limitados em termos de memória, processamento, bateria e conectividade, como os smartphones.

Para superar estes desafios, o MQTT utiliza uma abordagem de comunicação centrada em dados, denominada *Publish/Subscribe*. Desta forma, as

mensagens entregues aos destinatários são enviadas com base em seu interesse e conteúdo das mensagens, em vez do seu endereço de rede de destino, como acontece tradicionalmente ao se utilizar o protocolo TCP/IP.

Neste contexto, utilizou-se um *broker*, que é um intermediador da comunicação responsável por receber, filtrar, determinar quem está inscrito nos chamados "tópicos" e enviar as mensagens. Um tópico é uma chave que identifica o canal em que irão trafegar mensagens de uma determinada natureza. Por exemplo, no tópico "inavigs/joao/localizacao" irão trafegar mensagens referentes à localização do usuário "João". Todos os dispositivos inscritos irão receber as mensagens que trafegarem por este tópico.

Serviços INavigS

Conforme mostrado na Figura 4.11, o broker MQTT é encarregado de direcionar as mensagens que contém os dados obtidos pelo *smartphone* para um servidor que executa os serviços INavigS.

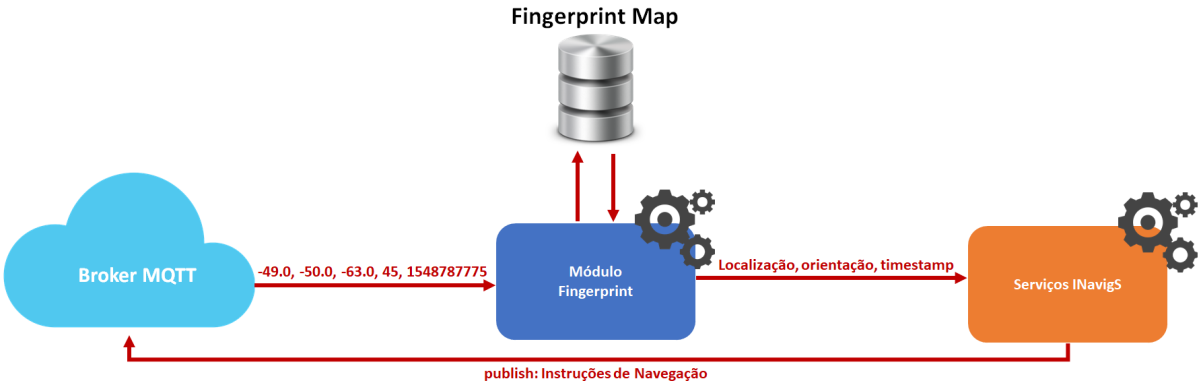

Figura 4.12: Comunicação entre os componentes.

A Figura 4.12 ilustra a organização dos componentes e como eles se comunicam. O módulo *fingerprint* executa o algoritmo KNN e ao receber os valores de RSSI medidos pelo *smartphone*, consulta o *Fingerprint Map* construído durante a fase *offline* para classificar as amostras e determinar a localização do usuário.

Ao obter a localização, o próximo passo é enviar os dados de contexto para a infraestrutura INavigS para que sejam calculadas as instruções de navegação. Logo que as informações chegam à infraestrutura, o parâmetro *Up-to-dateness* é calculado, com base na Equação 4.2 apresentada na Subseção 4.1.4. Se os dados de contexto apresentarem um grau que indique que ainda são válidos, as instruções de navegação

são calculadas e então enviadas de volta para o broker MQTT para que os dispositivos inscritos possam recebê-las. Caso contrário, a infraestrutura descarta as informações de contexto e aguarda até que novas informações cheguem até ela.

Todo o processo se repete até que o usuário chegue ao local pretendido ou cancele os serviços de navegação.

4.2 Considerações Finais

Neste capítulo foram apresentados os problemas enfrentados pela infraestrutura INavigS referentes à QoC, que motivaram este trabalho. Cada problema foi associado a parâmetros de QoC. Assim, três parâmetros foram destacados: *precision*, *completeness* e *up-to-dateness*.

Em seguida foram mostradas as técnicas utilizadas para abordar os problemas citados anteriormente e aplicar os parâmetros identificados.

Por fim, foi mostrado o cenário de execução da infraestrutura INavigS com os parâmetros e técnicas aplicados, bem como a arquitetura proposta e seus componentes. Espera-se que com o uso das técnicas propostas, a infraestrutura ofereça serviços de navegação com um maior nível de confiabilidade.

5 Avaliação e Experimentos

Este capítulo apresenta experimentos realizados no escopo deste trabalho para avaliar a proposta apresentada na Capítulo 4. Foram analisados aspectos relacionados à acurácia do método de localização proposto e um caso de uso para analisar a sua eficiência no cenário de navegação *indoor*.

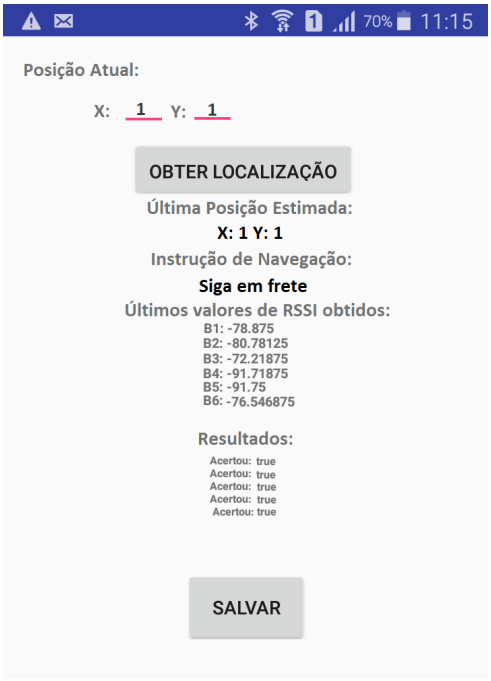

Figura 5.1: Aplicação desenvolvida para realizar os experimentos.

Para a realização do experimento foi desenvolvida uma aplicação que executa em dispositivos *Android*. Sua interface gráfica é mostrada na Figura 5.1. Nela, deve ser informada a posição onde o *smartphone* está localizado no momento. Ao ser pressionado o botão "Obter Localização", os valores de RSSI são lidos e exibidos, e o processo para calcular a localização é executado, conforme já descrito na Subseção 4.1.5. Em "Resultados" é exibido o valor *true* caso a localização estimada seja a mesma informada pelo usuário, e *false* em caso contrário.

Neste experimento foram utilizados os dispositivos mostrados na Tabela 5.1: um *smartphone* Samsung Galaxy S5 mini SM-G800H, com o sistema operacional *Android* na versão 6.0.1 para executar a aplicação mostrada anteriormente e *beacons* da marca *Radius Network*, modelo RadBeacon, compatível com a tecnologia BLE.

Tabela 5.1: Dispositivos Utilizados nos Experimentos

Tipo	Marca	Modelo	Sistema Operacional	Versão
Smartphone	Samsung	SM-G800H	Android	6.0.1
Beacon	Radius Network	RadBeacon Dot	N/A	N/A

Os *beacons* permitem configurações diferentes para definir o seu modo de funcionamento. A Tabela 5.2 mostra a configuração destes dispositivos no momento em que os experimentos foram executados. O parâmetro *Advertising rate* define quantas mensagens de publicidade são enviadas por segundo; *Transmit Power* indica a intensidade do sinal com que as mensagens são transmitidas. O formato das mensagens segue o padrão *iBeacon* que conta com informações sobre o UUID do *beacon* e o valor de RSSI em seu cabeçalho.

Tabela 5.2: Configuração dos *Beacons*

Parâmetro de Configuração	Valor
Advertising rate	10 anúncios por segundo
Transmit Power	+4 dBm
Protocolo de Transmissão	*iBeacon*

5.1 Experimento

O experimento foi realizado em um ambiente com uma área de aproximadamente 80 m^2, dividido em duas salas e um *hall*. Foram utilizados 6 *beacons* BLE distribuídos conforme mostrado na Figura 5.2.

5.1.1 Caso de Uso: Acurácia do método de localização proposto

Este experimento teve como objetivo analisar a capacidade em se obter a localização correta utilizando a técnica proposta.

O ambiente foi dividido em 14 posições, que são representadas pelos pontos destacados em vermelho mostrados na Figura 5.2. Durante a fase *offline*, procedimento já mostrado na Subseção 4.1.2, para cada um dos 6 *beacons* distribuídos no ambiente, foram lidos e armazenados 45 valores de RSSI, resultando em um *Fingerprint Map* com

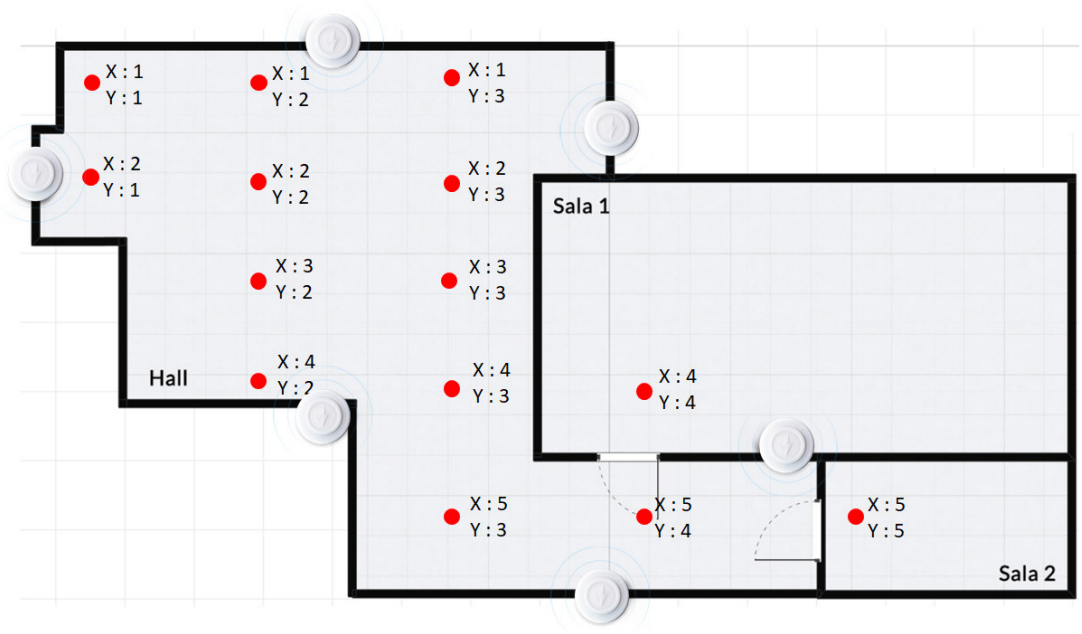

Figura 5.2: Cenário do caso de uso.

um total de 270 amostras. Cada posição foi identificada em forma de coordenada, com eixos x e y.

O experimento foi realizado de forma que o *smartphone* foi situado em cada uma das posições marcadas em vermelho na Figura 5.2, executando o aplicativo mostrado anteriormente na Figura 5.1. Antes de cada execução era informada manualmente a posição em que o *smartphone* estava localizado e esta era comparada com a localização obtida após a execução do processo. Foram realizadas 30 execuções das quais foi obtida a localização correta em 25 delas, totalizando uma taxa de acerto de 83,33%. Todas as execuções foram realizadas por um mesmo usuário.

5.1.2 Caso de Uso: Eficácia da técnica de localização proposta no cenário de uma aplicação de navegação *indoor*

O objetivo deste experimento foi avaliar a utilização da técnica de localização proposta, no cenário de uma aplicação de navegação *indoor*. O mesmo cenário do experimento anterior foi utilizado, entretanto dessa vez foi avaliada a capacidade da infraestrutura INavigS guiar o usuário pelo ambiente através das instruções de navegação fornecidas a partir da localização obtida pela técnica de proposta.

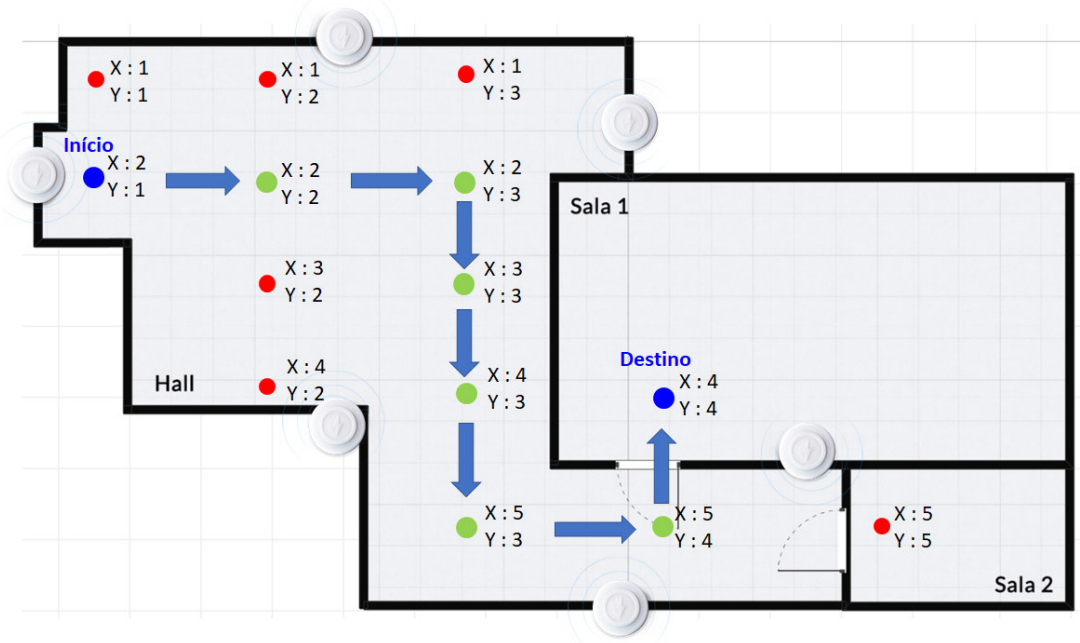

Figura 5.3: Cenário do caso de uso.

Durante o experimento, o *smartphone* foi situado inicialmente na posição X:2, Y:1, conforme mostrado na Figura 5.3. O objetivo era guiar o usuário do seu ponto de origem, que era o *Hall* – Posição X:2, Y:1 – até chegar à Sala 1 – Posição X:4, Y:4 – seguindo as instruções fornecidas pela infraestrutura INavigS, deslocando-se através das posições destacadas na cor verde.

Fatores como erros no cálculo da localização resultaram em instruções de navegação erradas que poderiam levar a desvios da rota a ser seguida. Portanto, foram analisadas duas situações, em que na primeira o usuário recebeu todas as instruções de navegação corretas e chegou ao destino sem desvio de rotas; e a segunda em que obteve-se a localização errada em algum ponto durante o percurso que o levou a um desvio da rota.

O procedimento foi executado 30 vezes, das quais houve sucesso em 26 das vezes, totalizando uma acurácia de 86%.

5.2 Considerações Finais

Este capítulo apresentou avaliações sobre a solução proposta através de casos de uso para analisar a acurácia da técnica de *fingerprinting* proposta.

Para isto, foi desenvolvido um protótipo de uma aplicação que adota a técnica de localização por *fingerprinting* para calcular a localização *indoor* e obter instruções de navegação *indoor* através da infraestrutura INavigS.

Foram realizados dois experimentos, o primeiro com objetivo de avaliar a acurácia da técnica de localização utilizada em termos do seu grau de acertos. Com este experimento foi possível detectar que a técnica proposta ofereceu um desempenho aceitável para calcular a localização *indoor*, com acurácia de 83,33%.

O segundo experimento consistiu em analisar a eficiência da técnica de localização proposta em um cenário de navegação *indoor*. Para isto foi avaliada capacidade da infraestrutura INavigS guiar um usuário a partir das informações de localização obtidas pelo módulo que executa a técnica de *fingerprinting*, obtendo-se sucesso em 26 vezes dos 30 testes executados, com uma acurácia de 86,67%.

6 Conclusão, Publicações e Trabalhos Futuros

Neste trabalho foram propostos métodos e técnicas para abordar problemas relacionado a QoC existentes em uma infraestrutura ciente de contexto que fornece serviços de navegação *indoor*, parte do escopo do projeto INavigS.

Três problemas foram identificados: baixa precisão nos dados de contexto sobre localização, alto tempo de atualização dos dados de contexto para a obtenção de instruções de navegação e ausência de dados sobre a orientação do usuário. Cada um destes problemas foi associados aos respectivos parâmetros de QoC: *Precision*, *Up-to-dateness* e *Completeness*.

Para abordar o problema referente ao parâmetro *Precision* foi utilizada a técnica de localização denominada *Fingerprinting*, que oferece uma maior precisão que a técnica baseada em Proximidade, atualmente utilizada pelo INavigS. Com o *fingerprinting* é possível obter a localização absoluta do usuário no ambiente. Em relação ao parâmetro *Completeness* foi utilizado um algoritmo de fusão de sensores que combinando dados dos sensores de Magnetômetro e Acelerômetro presentes na maioria dos *smartphones* para identificar a orientação do usuário e fornecer instruções de navegação mais completas. Por fim, para abordar o problema relacionado ao parâmero *Up-to-dateness* calculamos o tempo de validade das informações de contexto, descartando aquelas consideradas muito antigas para serem consumidas.

Realizamos experimentos e observamos que erros de operação de uma aplicação de navegação baseada na infraestrutura INavigS foram reduzidos. No cenário do nosso experimento a técnica de localização utilizada obteve uma acurácia de 83,33 %. Em uma aplicação de navegação *indoor* obteve-se uma acurácia de 86,67% em instruções de navegação corretas.

6.1 Publicações

Para divulgar os resultados obtidos ao longo desta pesquisa foi realizada a publicação de um artigo científico em periódico conforme mostrado a seguir:

- Sousa, T. S, Vale, S. B.; Teles, A. S.; Silva Neto, M. M. Quality of Context in an Indoor Navigation Infrastructure. Journal Of Convergence Information Technology (GYEONGJU). ISSN : 1975-9320 (Versão Impressa), 2233-9299 (Online). Publicado em 31 de Janeiro de 2019.
  Qualis CAPES (2013-2016): B1 em Ciência da Computação
  Situação: Publicado

6.2 Trabalhos Futuros

A partir deste trabalho, identificamos algumas possibilidades de trabalhos futuros como:

- Utilizar a técnica de localização *fingerprinting* com outros algoritmos de aprendizagem de máquina ou outras abordagens determinísticas ou probabilísticas e comparar com a técnica utilizada neste trabalho em termos de acurácia e desempenho;
- Combinar o uso de BLE com outras tecnologias para obter a localização *indoor*;
- Implementar uma aplicação de navegação *indoor* utilizando as técnicas propostas neste trabalho e a infraestrutura INavigS em um cenário real, como o campus da Universidade Federal do Maranhão;
- Explorar o uso da arquitetura proposta em aplicações focadas em fornecer usabilidade adequada a pessoas com deficiência;
- Adicionar serviços para representar o espaço *indoor* através de mapas.

Referências Bibliográficas

[1] WEISER, M. The computer for the 21st century. *Scientific american*, Nature Publishing Group, v. 265, n. 3, p. 94–104, 1991.

[2] ABOWD, G. D. et al. Towards a better understanding of context and context-awareness. In: *Proceedings of the 1st International Symposium on Handheld and Ubiquitous Computing*. London, UK, UK: Springer-Verlag, 1999. (HUC '99), p. 304–307. ISBN 3-540-66550-1. Disponível em: <http://dl.acm.org/citation.cfm?id=647985.743843>.

[3] BUCHHOLZ, T.; KUPPER, A.; SCHIFFERS, M. Quality of context: What it is and why we need it. *In Proceedings of the Workshop of the HP OpenView University Association: OVUA'03.*, 2003.

[4] GU, Y.; LO, A.; NIEMEGEERS, I. A survey of indoor positioning systems for wireless personal networks. *IEEE Communications Surveys Tutorials*, v. 11, n. 1, p. 13–32, First 2009. ISSN 1553-877X.

[5] MAUTZ, R. Overview of current indoor positioning systems. *Geodesy and Cartography*, v. 35, p. 18–22, 2009.

[6] ZAFARI, F. et al. An ibeacon based proximity and indoor localization system. *arXiv preprint arXiv:1703.07876*, 2017.

[7] BAHL, P.; PADMANABHAN, V. N. Radar: an in-building rf-based user location and tracking system. In: *Proceedings IEEE INFOCOM 2000. Conference on Computer Communications. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies (Cat. No.00CH37064)*. [S.l.: s.n.], 2000. v. 2, p. 775–784 vol.2. ISSN 0743-166X.

[8] FARAGHER, R.; HARLE, R. Location fingerprinting with bluetooth low energy beacons. *IEEE Journal on Selected Areas in Communications*, v. 33, n. 11, p. 2418–2428, Nov 2015. ISSN 0733-8716.

[9] TERAN, M. et al. Iot-based system for indoor location using bluetooth low energy. In: *2017 IEEE Colombian Conference on Communications and Computing (COLCOM)*. [S.l.: s.n.], 2017. p. 1–6.

[10] MEDEIROS, A. *INavigS: Uma infraestrutura de software ciente de contexto para navegação indoor*. Dissertação (Mestrado) — Universidade Federal do Maranhão, 2018.

[11] SATYANARAYANAN, M. Pervasive computing: vision and challenges. *IEEE Personal Communications*, v. 8, n. 4, p. 10–17, Aug 2001. ISSN 1070-9916.

[12] ADELSTEIN, F. et al. *Fundamentals of Mobile and Pervasive Computing*. Mcgraw-hill, 2004. (Telecom Engineering). ISBN 9780071412377. Disponível em: <https://books.google.com.br/books?id=f7WDu9z-x6AC>.

[13] ARAUJO, R. B. Computação ubíqua: Princípios, tecnologias e desafios. In: *Anais do XXI Simpósio Brasileiro de Redes de Computadores (SBRC 2003)*. [S.l.: s.n.], 2003. v. 8, p. 11–13.

[14] WANT, R. et al. The active badge location system. *ACM Transactions on Information Systems (TOIS)*, New York, NY, USA, v. 10, n. 1, p. 91–102, jan. 1992. ISSN 1046-8188. Disponível em: <http://doi.acm.org/10.1145/128756.128759>.

[15] SCHILIT, B. N.; THEIMER, M. M. Disseminating active map information to mobile hosts. *Netwrk. Mag. of Global Internetwkg.*, IEEE Press, Piscataway, NJ, USA, v. 8, n. 5, p. 22–32, set. 1994. ISSN 0890-8044. Disponível em: <http://dx.doi.org/10.1109/65.313011>.

[16] SCHILIT, B.; ADAMS, N.; WANT, R. Context-aware computing applications. In: *Proceedings of the 1994 First Workshop on Mobile Computing Systems and Applications*. Washington, DC, USA: IEEE Computer Society, 1994. (WMCSA '94), p. 85–90. ISBN 978-0-7695-3451-0. Disponível em: <http://dx.doi.org/10.1109/WMCSA.1994.16>.

[17] SALBER, D.; DEY, A. K.; ABOWD, G. D. The context toolkit: Aiding the development of context-enabled applications. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. New York, NY, USA: ACM, 1999. (CHI '99), p. 434–441. ISBN 0-201-48559-1. Disponível em: <http://doi.acm.org/10.1145/302979.303126>.

[18] SCHMIDT, A. Implicit human computer interaction through context. *Personal Technologies*, v. 4, n. 2, p. 191–199, Jun 2000. ISSN 1617-4917. Disponível em: <https://doi.org/10.1007/BF01324126>.

[19] CHEN, G.; KOTZ, D. *A Survey of Context-Aware Mobile Computing Research*. Hanover, NH, USA, 2000.

[20] JANG, S.; KO, E.-J.; WOO, W. Unified user-centric context: Who, where, when, what, how and why. *Personalized Context Modeling and Management for UbiComp Applications*, CEUR-WS, v. 149, n. 26-34, 2005.

[21] BALDAUF, M.; DUSTDAR, S.; ROSENBERG, F. A survey on context-aware systems. *International Journal of Ad Hoc and Ubiquitous Computing*, Inderscience Publishers, v. 2, n. 4, p. 263–277, 2007.

[22] AILISTO, H. et al. Structuring context aware applications: Five-layer model and example case. In: *Proceedings of the Workshop on Concepts and Models for Ubiquitous Computing*. [S.l.: s.n.], 2002. p. 1–5.

[23] KAPSAMMER, E.; SCHWINGER, W.; RETSCHITZEGGER, W. Bridging relational databases to context-aware services. In: *CAiSE Workshops (2)*. [S.l.: s.n.], 2005. p. 703–716.

[24] INDULSKA, J.; SUTTON, P. Location management in pervasive systems. In: *Proceedings of the Australasian Information Security Workshop Conference on ACSW Frontiers 2003 - Volume 21*. Darlinghurst, Australia, Australia: Australian Computer Society, Inc., 2003. (ACSW Frontiers '03), p. 143–151. ISBN 1-920682-00-7. Disponível em: <http://dl.acm.org/citation.cfm?id=827987.828003>.

[25] VIEIRA, V. et al. Uso e representação de contexto em sistemas computacionais. In: *Tópicos em Sistemas Interativos e Colaborativos*. São Carlos, São Paulo, Brasil: UFSCAR, 2006. p. 127–166.

[26] RAYCHOUDHURY, V. et al. Middleware for pervasive computing: A survey. *Pervasive and Mobile Computing*, v. 9, n. 2, p. 177 – 200, 2013. ISSN 1574-1192. Special Section: Mobile Interactions with the Real World. Disponível em: <http://www.sciencedirect.com/science/article/pii/S1574119212001113>.

[27] SILVA, C.; DANTAS, M. Quality-aware context provider: A filtering approach to context-aware systems on ubiquitous environment. In: IEEE. *Wireless and Mobile Computing, Networking and Communications (WiMob), 2013 IEEE 9th International Conference on*. [S.l.], 2013. p. 422–429.

[28] HENRICKSEN, K.; INDULSKA, J. Modelling and using imperfect context information. In: *IEEE Annual Conference on Pervasive Computing and Communications Workshops, 2004. Proceedings of the Second*. [S.l.: s.n.], 2004. p. 33–37.

[29] KRAUSE, M.; HOCHSTATTER, I. Challenges in modelling and using quality of context (qoc). *Mobility aware technologies and applications*, Springer, p. 324–333, 2005.

[30] MANZOOR, A.; TRUONG, H.-L.; DUSTDAR, S. Quality of context: models and applications for context-aware systems in pervasive environments. *The Knowledge Engineering Review*, Cambridge University Press, v. 29, n. 2, p. 154–170, 2014.

[31] SHEIKH, K.; WEGDAM, M.; SINDEREN, M. van. Quality-of-context and its use for protecting privacy in context aware systems. *JSW*, v. 3, n. 3, p. 83–93, 2008.

[32] NAZÁRIO, D. C. et al. *Cuida: um modelo de conhecimento de qualidade de contexto aplicado aos ambientes ubíquos internos em domicílios assistidos*. Tese (Doutorado) — Universidade Federal de Santa Catarina, 2015.

[33] KIM, Y.; LEE, K. A quality measurement method of context information in ubiquitous environments. In: *2006 International Conference on Hybrid Information Technology*. [S.l.: s.n.], 2006. v. 2, p. 576–581.

[34] KIM, J.-E. et al. Navigating visually impaired travelers in a large train station using smartphone and bluetooth low energy. In: *Proceedings of the 31st Annual ACM Symposium on Applied Computing*. New York, NY, USA: ACM, 2016. (SAC '16), p. 604–611. ISBN 978-1-4503-3739-7. Disponível em: <http://doi.acm.org/10.1145/2851613.2851716>.

[35] DUSTDAR, S.; MANZOOR, A. *Quality of Context in Pervasive Systems: Models, Techniques, and Applications*. Tese (Doutorado) — Ph. D Thesis, Technical University of Vienna, 2010.

[36] WANG, R. Y.; STRONG, D. M. Beyond accuracy: What data quality means to data consumers. *J. Manage. Inf. Syst.*, M. E. Sharpe, Inc., Armonk,

NY, USA, v. 12, n. 4, p. 5–33, mar. 1996. ISSN 0742-1222. Disponível em: <http://dx.doi.org/10.1080/07421222.1996.11518099>.

[37] HUEBSCHER, M. C.; MCCANN, J. A. Adaptive middleware for context-aware applications in smart-homes. In: *Proceedings of the 2Nd Workshop on Middleware for Pervasive and Ad-hoc Computing*. New York, NY, USA: ACM, 2004. (MPAC '04), p. 111–116. ISBN 1-58113-951-9. Disponível em: <http://doi.acm.org/10.1145/1028509.1028511>.

[38] TALAVERA, L. E. et al. The mobile hub concept: Enabling applications for the internet of mobile things. In: *2015 IEEE International Conference on Pervasive Computing and Communication Workshops (PerCom Workshops)*. [S.l.: s.n.], 2015. p. 123–128.

[39] NEWMAN, N. Apple ibeacon technology briefing. *Journal of Direct, Data and Digital Marketing Practice*, v. 15, n. 3, p. 222–225, Jan 2014. ISSN 1746-0174. Disponível em: <https://doi.org/10.1057/dddmp.2014.7>.

[40] PNT. *Official U.S. government information about the Global Positioning System (GPS) and related topics*. 2018. Disponível em: <https://www.gps.gov/systems/gps/performance/accuracy/>.

[41] MAUTZ, R. Overview of current indoor positioning systems. *Geodezija ir kartografija*, v. 35, n. 1, p. 18–22, 2009.

[42] KAJIOKA, S. et al. Experiment of indoor position presumption based on rssi of bluetooth le beacon. In: *2014 IEEE 3rd Global Conference on Consumer Electronics (GCCE)*. [S.l.: s.n.], 2014. p. 337–339. ISSN 2378-8143.

[43] CORREA, A. et al. A review of pedestrian indoor positioning systems for mass market applications. *Sensors*, v. 17, n. 8, 2017. ISSN 1424-8220. Disponível em: <http://www.mdpi.com/1424-8220/17/8/1927>.

[44] FARID, Z.; NORDIN, R.; ISMAIL, M. Recent advances in wireless indoor localization techniques and system. *Journal of Computer Networks and Communications*, Hindawi Publishing Corporation, v. 2013, 2013. ISSN 2090-7141.

[45] CORDEIRO, J. T. P. *Técnicas de localização Indoor*. Dissertação (Mestrado) — UNIVERSIDADE DE TRAS-OS-MONTES E ALTO DOURO, http://hdl.handle.net/10348/5921, 2016.

[46] HIGHTOWER, J.; BORRIELLO, G. Location sensing techniques. *IEEE Computer*, v. 34, n. 8, p. 57–66, 2001.

[47] GU, Y.; LO, A.; NIEMEGEERS, I. A survey of indoor positioning systems for wireless personal networks. *IEEE Communications Surveys Tutorials*, v. 11, n. 1, p. 13–32, First 2009. ISSN 1553-877X.

[48] CHANDEL, V. et al. Inloc: An end-to-end robust indoor localization and routing solution using mobile phones and ble beacons. In: *2016 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*. [S.l.: s.n.], 2016. p. 1–8.

[49] ZHOU, Y. An efficient least-squares trilateration algorithm for mobile robot localization. In: *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*. [S.l.: s.n.], 2009. p. 3474–3479. ISSN 2153-0858.

[50] CHERAGHI, S. A.; NAMBOODIRI, V.; WALKER, L. Guidebeacon: Beacon-based indoor wayfinding for the blind, visually impaired, and disoriented. In: *2017 IEEE International Conference on Pervasive Computing and Communications (PerCom)*. [S.l.: s.n.], 2017. p. 121–130.

[51] AHMETOVIC, D. et al. Navcog: A navigational cognitive assistant for the blind. In: *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services*. New York, NY, USA: ACM, 2016. (MobileHCI '16), p. 90–99. ISBN 978-1-4503-4408-1. Disponível em: <http://doi.acm.org/10.1145/2935334.2935361>.

[52] Han, D. et al. Building a practical wi-fi-based indoor navigation system. *IEEE Pervasive Computing*, v. 13, n. 2, p. 72–79, Apr 2014. ISSN 1536-1268.

[53] NEISSE, R.; WEGDAM, M.; SINDEREN, M. v. Trustworthiness and quality of context information. In: *2008 The 9th International Conference for Young Computer Scientists*. [S.l.: s.n.], 2008. p. 1925–1931.

[54] Corradi, A.; Fanelli, M.; Foschini, L. Adaptive context data distribution with guaranteed quality for mobile environments. In: *IEEE 5th International Symposium on Wireless Pervasive Computing 2010*. [S.l.: s.n.], 2010. p. 373–380.

[55] ORUJOV, F.; MASKELIUNAS, R. Comparative analysis of the indoor positioning algorithms using bluetooth low energy beacons. In: *2016 CEUR Workshop Proceedings*. [S.l.: s.n.], 2016.

[56] HONKAVIRTA, V. et al. A comparative survey of wlan location fingerprinting methods. In: *2009 6th Workshop on Positioning, Navigation and Communication*. [S.l.: s.n.], 2009. p. 243–251.

[57] HE, S.; CHAN, S. . G. Wi-fi fingerprint-based indoor positioning: Recent advances and comparisons. *IEEE Communications Surveys Tutorials*, v. 18, n. 1, p. 466–490, Firstquarter 2016. ISSN 1553-877X.

[58] MOURãO, H. O. H. Localização de dispositivos móveis em redes wifi usando variação da potência de transmissão e knn. *Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*, p. 385–396, 2013.

[59] ABADI, M. J. et al. Improving heading accuracy in smartphone-based pdr systems using multi-pedestrian sensor fusion. *Electr. Eng*, v. 188, p. 9–35, 2013.

[60] MANZOOR, A.; TRUONG, H.-L.; DUSTDAR, S. On the evaluation of quality of context. In: SPRINGER. *European Conference on Smart Sensing and Context*. [S.l.], 2008. p. 140–153.