UNIVERSIDADE FEDERAL DE SANTA CATARINA
CAMPUS FLORIANÓPOLIS
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Victoria Botelho Martins

Service placements, fog and mobility: a study towards the state of the art

Florianópolis
2023

Victoria Botelho Martins

Service placements, fog and mobility: a study towards the state of the art

Dissertação submetida ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Santa Catarina para a obtenção do título de mestre em Ciência da Computação.
Supervisor: Prof. Douglas Dyllon Jeronimo de Macedo, Dr.

Florianópolis
2023

Ficha de identificação da obra elaborada pelo autor,
através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

Martins, Victoria Botelho
   Service placements, fog and mobility: a study towards the state of the art / Victoria Botelho Martins ; orientador, Douglas Dyllon Jeronimo de Macedo, 2023.
   121 p.

   Dissertação (mestrado) - Universidade Federal de Santa Catarina, Centro Tecnológico, Programa de Pós-Graduação em Ciência da Computação, Florianópolis, 2023.

   Inclui referências.

   1. Ciência da Computação. 2. Internet of Things. 3. Fog Computing. 4. Mobilidade. 5. Sistemas de Localização. I. Macedo, Douglas Dyllon Jeronimo de. II. Universidade Federal de Santa Catarina. Programa de Pós-Graduação em Ciência da Computação. III. Título.

Victoria Botelho Martins

Service placements, fog and mobility: a study towards the state of the art

O presente trabalho em nível de mestrado foi avaliado e aprovado por banca examinadora composta pelos seguintes membros:

Certificamos que esta é a **versão original e final** do trabalho de conclusão que foi julgado adequado para obtenção do título de mestre em Ciência da Computação.

GOV.BR
Documento assinado digitalmente
PATRICIA DELLA MEA PLENTZ
Data: 01/06/2023 19:08:58-0300
CPF: ***.243.700-**
Verifique as assinaturas em https://v.ufsc.br

Coordenação do Programa de
Pós-Graduação em Ciência da
Computação

ICP-Edu
Documento assinado digitalmente
Douglas Dyllon Jeronimo de Macedo
Data: 01/06/2023 10:20:38-0300
CPF: ***.987.372-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Douglas Dyllon Jeronimo de Macedo,
Dr.
Supervisor:

Florianópolis, 2023.

AGRADECIMENTOS

Agradeço ao Professor Douglas por ter me aceitado como aluna e também ao Laércio, André e à Luciele pelas iluminações durante o processo.

Agradecimentos à Universidade Federal de Santa Catarina, Coordenação de Aperfeiçoamento de Pessoal de Nível Superior e aos laboratórios LabSec e LEA pelas bolsas ofertadas enquanto estudante e programadora.

Agradecimentos à comunidade open source por fornecer ferramentas que facilitam o trabalho como programadores e pesquisadores, são eles: MermaidJS, PlantUML, Heatmapper.ca, iFogSim, Draw.io, GNUPLOT, LaTex.

À comunidade de todas as pessoas que indiretamente contribuíram com o desenvolvimento deste trabalho, elas importam.

E por fim, mas não menos importante, agradeço diretamente à vovó e ao vovô pelas orações direcionadas à mim, mesmo quando fingia que estava tudo bem. Dona "Grória" também, sempre por ter me dado suporte e me entendido quando fingia que estava tudo bem.

*Desta vez, nenhum programadorx foi ferido no processo.

“Everlasting love is ever-growing
Hang on to what you have and let it grow
Everlasting love is ever-dying
It’s in the past, you have to let it go
(The Shooting Star - Gojira)

RESUMO

Com o advento da Internet das Coisas (IoT), sistemas de posicionamento tem desbravado cenários além de fins navais e bélicos. Hoje, tais sistemas podem ser apoiados pelo conjunto de protocolos TCP/IP, acelerando sua implementação em diferentes cenários em vários setores da sociedade. Por definição, IoT pode ser descrito como uma rede de objetos físicos e heterogêneos conectados à Internet. Devido à ubiquidade destes dispositivos que podem ser denominados como borda, e de sua conexão com a Internet, estes contribuem para a implementação de sistemas de posicionamento em vários contextos. Tais aplicações integram hardware e software para fornecer continuamente a localização dos dispositivos. Cenários internos, como hospitais e demais edifícios, por serem cercados por concreto, requerem tecnologias e métodos de localização diferentes daqueles usados em cenários externos, como sistemas de navegação por satélite. Além disso, certos contextos, como museus e aeroportos inteligentes, podem ter variáveis dinâmicas, como o número de dispositivos detectáveis, resultando em alta latência de rede para arquiteturas baseadas em nuvens centrais. O crescente desenvolvimento de IoT tem motivado a distribuição do poder computacional para a borda das arquiteturas, reduzindo o tempo de requisição-resposta e resultando em menos processamento em data centers de alto desempenho. Entre a nuvem e os usuários finais, existe a névoa (fog), um paradigma computacional que visa dimensionar os recursos computacionais para mais próximo dos usuários e melhorar a qualidade do serviço reduzindo a latência e aumentando o conhecimento do serviço (awareness). No entanto, ao se aproximar do poder de processamento da borda da rede, ainda utilizam-se estratégias tradicionais de alocação de serviços. Em ambientes móveis dinâmicos, um estudo avaliando o desempenho do sistema pode beneficiar seu desempenho e compreender suas particularidades além de modelos de mobilidade e posicionamento de nodos de névoa. Desta forma, o este estudo teve como objetivo investigar se as alocações tradicionais de serviços de localização podem ser tão agnósticas de aplicação espacial (space-application-agnostic) em IPS (sistemas de posicionamento interno) quanto em OPS (sistemas de posicionamento externo), ou mesmo se poderiam ser equivalentes. Para diversificar o estudo, experimentos foram feitos usando padrões de mobilidade aleatórios adaptados em simulações. Usando as metodologias propostas e o software de simulação iFogSim v2, foi possível explorar posicionamentos clustered e edge-ward em sistemas de posicionamento indoor e outdoor e desenvolver um conjunto de modelos de mobilidade com base na equivalência dos posicionamentos. A pesquisa mostrou que as estratégias tradicionais de posicionamento não são equivalentes ao IPS, provando que não são estratégias comparáveis e tendem a ter pior desempenho para migrações indoor. Além disso, foi determinado que a falta de definição do que se constitui ambientes internos e consciência de contexto (context-awareness) é um fator importante para o IPS, uma vez que os modelos de mobilidade usados tendem a ser aleatórios na literatura. Além das descobertas mencionadas e dos benchmarks padronizados desenvolvidos, foi atestado que as colocações em cluster tendem a estar mais próximas do agnosticismo de espacial promovido e almejado pela comunidade de código aberto.

Palavras-chave: Internet das Coisas, Fog Computing, Mobilidade, Sistemas de Localização.

ABSTRACT

As most of the scenarios for positioning systems were outdoors and were historically used for naval and warfare purposes, positioning systems were satellite-based. However, with the advent of the Internet of Things (IoT), these systems can be supported by the TCP/IP protocol suite, which is able to accelerate their implementation in different scenarios. By definition, the IoT can be described as a network of physical and heterogeneous objects connected to the Internet, and nowadays it is part of a wide range of intelligent solutions, ranging from essential scenarios such as agriculture and power grids to other solutions closer to end users, which can be smartphones, for example. Because of their ubiquity, these devices, which can be referred to as edge devices, contribute to the implementation of positioning systems in various contexts. These applications integrate hardware and software to continuously provide the location of devices. Indoor scenarios, such as hospitals and other buildings, because their spaces are surrounded by concrete, require technologies and location methods different from those used in outdoor scenarios, like GPS and other global navigation satellite systems. Furthermore, certain contexts, such as museums and smart airports, may have dynamic variables, such as the number of discoverable devices, resulting in high network latency for architectures based on Core Clouds. The growing development of IoT has motivated the distribution of computational power to the edge of architectures, reducing the request-response time and resulting in less processing in high-performance data centers. Between the cloud and the users, there is the fog, a computing paradigm that aims to scale computational resources closer to the users and enhance the quality of service by reducing latency and increasing service awareness. However, as it brings processing power closer to the edge of the network, it uses traditional service placement strategies. Nevertheless, in dynamic mobile environments, a study evaluating the system's performance could benefit their performance and comprehend their particularities (contextness) aside from fog node positioning and mobility models. This way, the present work aimed to investigate if traditional location service placements could be as space application agnostic in IPS (indoor positioning systems) as they are in OPS (outdoor positioning systems), or even if they could be equivalent. To diversify the study, experiments were made using adapted random mobility patterns supported by simulations. Using the proposed methodologies and the simulation software iFogSim v2, it was possible to explore clustered and edge-ward placements in both indoor and outdoor positioning systems and develop a set of mobility models based on the equivalence of the positionings. Our research showed the traditional placement strategies are not equivalent to IPS, proving they are not comparable strategies and tend to have worse performance for migration indoors. Also, it was determined that a lack of definition of what constitutes indoor environments and context-awareness is an important factor for IPS since the mobility models used tend to be randomized in the literature. Besides the aforementioned findings and the standardized benchmarks developed, it was attested that clustered placements tend to be closer to the space application agnosticism promoted by the open source community.

Keywords: Internet of Things, Fog Computing, Mobility, Location Systems.

LIST OF FIGURES

Figure 1 – Positioning system classification from (FARAHSARI et al., 2022) . . 25
Figure 2 – Wifi fingerprint based on (ZHAO et al., 2018) 28
Figure 3 – IoT Computing Paradigms (YOUSEFPOUR et al., 2019) 29
Figure 4 – Standard fog computing architecture from (IORGA et al., 2018) . . . 33
Figure 5 – Topology using clustered placement 34
Figure 6 – Application flowchart using vertical placement 35
Figure 7 – Topology using vertical placement 35
Figure 8 – Categories of mobility models used ind MANETs from (BAI; HELMY, 2004) 40
Figure 9 – Traveling pattern of an MN using the Random Waypoint mobility model from (ARIYAKHAJORN; WANNAWILAI; SATHITWIRIYAWONG, 2006) 40
Figure 10 – Research methodology procedures (Stages 1 - 4) 62
Figure 11 – Research methodology procedures (Stages 5 - 7) 63
Figure 12 – Conceptual model of architectures 64
Figure 13 – Migration intercluster in clustered approach 65
Figure 14 – Migration intracluster in clustered approach 65
Figure 15 – Application model for the indoor case studies based on the modules present in (PEŠIĆ et al., 2018) and (PEŠIĆ et al., 2019) 65
Figure 16 – Location of the entries and boarders of the proposed ENS scenario 66
Figure 17 – Moving 5 users and 25 in the ENS scenario 66
Figure 18 – Heatmaps for 5 users and 25 in the ENS scenario 66
Figure 19 – Location of the entries and boarders of the proposed INE scenario . 67
Figure 20 – Moving 5 users and 25 in the INE scenario 67
Figure 21 – Heatmaps for 5 users and 25 in the INE scenario 67
Figure 22 – 16 gateways with 3 proxies on the left and 4 on the right in the ENS scenario 68
Figure 23 – 12 gateways with 2 left and 3 right proxies in the ENS scenario . . . 68
Figure 24 – 8 gateways with 2 left and 3 right proxies in the ENS scenario 68
Figure 25 – 4 gateways with 1 proxy on the left and 2 on the right in the ENS scenario 69
Figure 26 – 16 gateways with 3 proxies on the left and 4 on the right in the INE scenario 69
Figure 27 – 12 gateways with 2 left and 3 right proxies in the INE scenario 69
Figure 28 – 8 gateways with 2 left and 3 right proxies in the INE scenario 70
Figure 29 – 4 gateways with 1 proxy on the left and 2 on the right in the INE scenario 70

Figure 30 – 20 nodes with 1 region on the left and 2 on the right in the ENS scenario on E2 71
Figure 31 – 20 nodes with 3 regions on the left and 4 on the right in the ENS scenario on E2 71
Figure 32 – 20 nodes with 1 region on the left and 2 on the right in the INE E2 scenario 72
Figure 33 – 20 nodes with 3 regions on the left and 4 on the right in the INE E2 scenario 72
Figure 34 – Application Model used in OPS experiment 72
Figure 35 – Eight, twelve and twenty eight clusters configurations 73
Figure 36 – A user path from south to east 73
Figure 37 – Configurations for OPS environments (T1 on the left and T2 on the right) 73
Figure 38 – Configurations for IPS environments (T1 on the left and T2 on the right) 73
Figure 39 – Cross-area mobility pattern in outdoor (left) and indoor (right) environments 74
Figure 40 – Line-through mobility pattern in outdoor (left) and indoor (right) environments 74
Figure 41 – Inline mobility pattern in indoor and outdoor environments 74
Figure 42 – ALD average per configuration in ENS with 5 and 25 users in E1 . . 76
Figure 43 – Average NU per configuration in ENS with 5 and 25 users in E1 . . . 77
Figure 44 – Average ALD per configuration in INE with 5 (left) and 25 (right) users in E1 78
Figure 45 – Average NU per configuration in INE with 5 users in E1 79
Figure 46 – Average ALD per configuration in ENS with 5 users on E2 80
Figure 47 – Average of NU in ENS with 5 and 25 users on E2 81
Figure 48 – Average ALD per configuration in INE with 5 and 25 users in E2 . . . 82
Figure 49 – Average of NU in INE with 5 and 25 users in E2 83
Figure 50 – Application loop delay and network usage per number of clusters in eight users scenario 85
Figure 51 – Migration time per number of clusters in eight users scenario 85
Figure 52 – Application loop delay and network usage per cluster in the fifty users scenario 85
Figure 53 – Migration time per number of clusters in fifty users scenario 86
Figure 54 – Application Loop Delay for IPS based on mobility patterns 87
Figure 55 – IPS Migration Time based on mobility patterns 87
Figure 56 – IPS Network Usage based on mobility patterns 88
Figure 57 – Application Loop Delay for OPS based on mobility patterns 88
Figure 58 – Migration Time for OPS based on mobility patterns 89

Figure 59 – Network Usage for OPS based on mobility patterns 89

LIST OF TABLES

Table 1 – Comparison of wireless radio frequency IPS technologies 27
Table 2 – Comparison of fog simulators regarding academia aspects 38
Table 3 – Comparison of fog simulators regarding development aspects 39
Table 4 – Synthesis of search elements 41
Table 5 – Relation between the search expression and the results obtained from the sources 41
Table 6 – Criteria used in the phases 42
Table 7 – Comparison of articles and the research proposal 45
Table 8 – Search Characterization Synthesis 46
Table 9 – Nodes simulation parameters 53
Table 10 – Latency simulation parameters 53
Table 11 – Simulation parameters for application modules 53
Table 12 – Correlation of fog nodes in experiment one 55
Table 13 – Correlation of fog nodes in experiment two 56
Table 14 – Relation of users in eight-users dataset 58
Table 15 – Simulation parameters for the devices 58
Table 16 – Relation of interactions, meter and environments of scenario one . . 60
Table 17 – Relation of interactions, meter and environments of scenario two . . . 60
Table 18 – Relation of interactions, meter and environments of scenario three . . 61

LIST OF ABBREVIATIONS AND ACRONYMS

FC	Fog Computing
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
IoT	Internet of Things
IPS	Indoor Positioning System
MQTT	Message Queuing Telemetry Transport
MS	Microservice
PS	Positioning System
QoS	Quality of Service
SLA	Service Level Agreements
UFSC	Universidade Federal de Santa Catarina
VANET	Vehicular Ad-hoc Network
VM	Virtual Machine

CONTENTS

1 INTRODUCTION 17
1.1 PROBLEM 19
1.2 MOTIVATION 20
1.3 OBJECTIVES 21
1.4 CONTRIBUTIONS 21
1.5 SCOPE DELIMITATION 22
1.6 DOCUMENT OUTLINE 23
2 BACKGROUND 24
2.1 POSITIONING VS LOCATION VS NAVIGATION VS TRACKING SYSTEMS 24
2.2 INDOOR POSITIONING SYSTEMS 25
2.2.1 Wireless Local Area Network IPS 27
2.3 ENABLING IOT COMPUTING PARADIGMS OR IOT-ENABLED COMPUTING PARADIGMS 28
2.4 FOG COMPUTING 31
2.4.1 Fog computing vs. Edge computing 31
2.4.2 Standard fog architecture 32
2.5 STRATEGIES FOR SERVICE PLACEMENT 32
2.5.1 Clustering placement 33
2.5.2 Edgewards placement 34
2.5.3 Clustered and Edge-ward placements in OPS 34
2.6 AWARENESS VS AGNOSTICISM 35
2.7 VIRTUAL EXPERIMENTATION ENVIRONMENTS 36
2.7.1 Indoor Positioning Systems simulators 37
2.7.2 Fog computing simulators supporting mobile nodes 37
2.8 MOBILITY MODELS 39
2.8.1 Random waypoint 40
3 BIBLIOGRAPHIC REVIEW 41
3.1 RESEARCH PLANNING 41
3.2 PRIMARY STUDIES SELECTION 42
3.2.1 Phase 0 42
3.2.2 Phase 1 42
3.3 RELATED WORKS 43
3.4 RESULT ANALYSIS 44
3.5 DISCUSSION 45
4 IPS PROPOSAL AND RESEARCH METHODOLOGY 46
4.1 METHODOLOGICAL PROCEDURES 46

4.2 SIMULATION 47
4.2.1 Architectures, topologies, and conceptual models 48
4.2.2 Vertical placement algorithm 48
4.2.3 Horizontal placement algorithm 50
4.2.4 Migration event 50
4.3 SIMULATION PARAMETERS 50
4.4 APPLICATION MODEL 50
4.5 DATA 52
4.6 FOG CONFIGURATIONS 53
4.7 CONTEXTS 54
4.7.1 Indoor Positioning Systems scenarios 54
4.7.1.1 Department of Sanitary Engineering (ENS) 55
4.7.1.2 Department of Informatics and Statistics (INE) 55
4.7.1.3 Experiment one - E1 55
4.7.1.4 Experiment two - E2 56
4.7.2 Outdoor Positioning Systems scenarios 56
4.7.2.1 Fog nodes configurations 57
4.7.2.2 Users mobility 57
4.7.3 Comparison of indoor to outdoor environment placement strategies 58
4.7.3.1 Fog nodes configurations 58
4.7.3.2 Scenarios 59
4.8 DISCUSSION 60
5 EXPERIMENTAL RESULTS AND EVALUATION 75
5.1 METRICS 75
5.2 EVALUATION OF INDOOR POSITIONING SYSTEMS EXPERIMENT 75
5.2.1 Scenario ENS + E1 75
5.2.1.1 ALD benchmarks 75
5.2.1.2 NU benchmarks 76
5.2.2 Scenario INE + E1 77
5.2.2.1 ALD benchmarks 78
5.2.2.2 NU benchmarks 79
5.2.3 Scenario ENS + E2 79
5.2.3.1 ALD benchmarks 80
5.2.3.2 NU benchmarks 80
5.2.4 Scenario INE + E2 81
5.2.4.1 ALD benchmarks 81
5.2.4.2 NU benchmarks 82
5.2.5 Discussion 83

5.3 EVALUATION OF OUTDOOR POSITIONING SYSTEMS EXPERIMENT 84
5.3.1 Eight users scenario 84
5.3.2 Fifty users scenario 85
5.3.3 Discussion 86
5.4 EVALUATION OF OPS-IPS ENVIRONMENT 86
5.4.1 Indoor environment 86
5.4.2 Outdoor environment 88
5.4.3 Discussion 89
5.5 CHALLENGES AND LIMITATIONS 90
6 CONCLUSION AND FUTURE WORK 91
REFERENCES 94
APPENDIX A – MODIFIED RANDOM WAYPOINT ALGORITHM MOBILITY MODEL FOR IPS 105
APPENDIX B – CLUSTER FORMATION ALGORITHM 112
ANNEX A – DEFAULT RANDOM WAYPOINT ALGORITHM MOBILITY MODEL FOR OPS FROM THE SIMULATOR . . . 114

1 INTRODUCTION

The rapid evolution of Internet of Things (IoT) devices has significantly increased the volume, speed, and variety of data (CISCO, 2015). As a result, the global IoT market is expected to exceed $1 trillion in four years (SAMANN; ZEEBAREE; ASKAR, 2021). In order to avoid costs, service providers have endeavored to follow this evolution by providing Quality of Service (QoS) in accordance with the Service Level Agreement (SLA) provided (SAMANN; ZEEBAREE; ASKAR, 2021). In order to ease SLA flaws, data reduction solutions have been developed on the edge of the network (PIOLI et al., 2022). However, as it has been noticed, the computing processing has been shifted towards the localities of the end users.

For a long time, the main infrastructure sustaining IoT ecosystems has been the core clouds (LAN et al., 2019). However, due to the increasing demand for performance in different scenarios, other solutions taking into account the IoT environment have been considered (COUTINHO et al., 2018b). A computing paradigm that has contributed to the improvement of services provided by the network is fog computing (FC) (LAN et al., 2019). This paradigm, capable of providing a response time window closer to real time, also reducing latency in IoT applications (CISCO, 2015), has been expanding the possibilities of architectures by bringing more computational power to the edge of the network (BONOMI et al., 2012).

Through the TCP/IP suite of protocols, the IoT has provided modern solutions using the concept of a unified framework for sharing information across different platforms and, therefore, possibly contexts. Each "thing" of an Internet of Things application needs a physical representation, connection abilities, and a unique name or address besides the computing unit. While these few and low requirements expand application possibilities, they also increase device heterogeneity (PALLEWATTA; KOSTAKOS; BUYYA, 2019).

Approaching IoT devices, for example, (which can be understood as edge devices), different paradigms have been studied to extend the power and maintainability of IoT environments. In the context of assisted living environments for healthcare, studies for mitigating IoT data offloading on the network have been studied (MACEDO, D. D. de; VON WANGENHEIM; DANTAS, 2015). Devices in IoT environments cannot only be connected to the Internet but also can be available through Wireless technologies like Wi-Fi and Bluetooth (FIROUZI et al., 2020). These devices acquire new interaction methods with the environment as they can sense, monitor and even control systems like smart homes and even autonomous vehicles. The concept of smart environments can also be applied to small and medium environments. In positioning systems (PS), for example, location nodes are responsible for providing the position of the mobile nodes. Besides having the computing power to estimate and calculate their position, their location plays a major role in this service provision. Studies such as (MARTINS et al., 2022), aimed to develop an algorithm to create clusters based on node range capacity.

As benefits, IoT provides more information for business decision making (FIROUZI et al., 2020), after all these devices, such as *smartphones*, are closer to your customers. However, due to the diffusion of these devices, it is estimated that by the next decade of 2030, the number of these devices will triple (SAMANN; ZEEBAREE; ASKAR, 2021). Thus, the number of devices will be needed in increasingly efficient, secure and scalable ways (GUBBI et al., 2013).

Massive data can be observed in sensible applications, such as healthcare environments. Yet, optimizing the amount of data generated is not only a data modeling problem but also a concern for data reduction (MACEDO, D. D. de; VON WANGENHEIM; DANTAS, 2015), independent of pre- or post-processing events. However, non-sensible applications can be observed in the literature. And since these contexts may have different priorities, at the edge level, specialized approaches (MACEDO, D. D. J. de et al., 2019) have been developed to increase management levels.

Nowadays, it is possible to encounter various smart environments. In the literature, a smart environment can be defined by the implementation of IoT (FARAHSARI et al., 2022). Since the edge devices in the IoT ecosystem just need to be connected to the internet, the ease of implementation of IoT applications is one of the factors that accelerated its rise (PALLEWATTA; KOSTAKOS; BUYYA, 2019). This effect can be seen at different scales: smart fridges, homes, airports, parking lots, and even the concept of smart cities and intelligent transportation systems.

Fog extends the cloud towards the devices (BONOMI et al., 2012) and the main motif is to aid the services provided. Offering a more generic alternative that natively supports large amounts of traffic and allows resources to be anywhere along the edge-to-cloud continuum (YOUSEFPOUR et al., 2019). To add a layer between things (the edge) and the cloud is to increase responsiveness, privacy protection, and location awareness. As fog means to alter the IoT infrastructure, there have been developed different computing paradigms within it based on application purposes. Vehicular Ad-hoc Networks (VANETs) are vivid emerging examples of IoT (MAAD HAMDI et al., 2020).

Open source fog simulators, emulators and testbeds are endorsed by companies and consortiums. The EdgeX Foundry (JOHN et al., 2021) is one example of open source platform that aims to standardize Industrial IoT. Nowadays, this project been supported by powerful companies such as Intel. Also registered by the Linux Foundation, the KubeEdge project brings containers architecture to the edge (YANG et al., 2022).

Since mobility is usually thought of as applied to outdoor environments, service placements, topologies, configurations, and experimentation have been used in larger geographical environments, like smart cities (LAI et al., 2018). However, as each IoT application has its own demands, a study evaluating the performance of service placements (MARTINS et al., 2022) in fog-based IPS based on mobility patterns as well as user interactions is yet to be developed. A lack of research in these contexts can result in inadequate architectures for system maintainability.

1.1 PROBLEM

Mobile devices are non-fixed sources that have trajectories that are not necessarily predictable. Thus, mobility is a subject of research in different systems. This element affects, for example, mobile networks, which depend mainly on the positioning and signal level of their towers, as well as ad-hoc networks that adapt their topologies dynamically. Vehicular networks (VANETs) also use the same point-to-point principles as ad hoc networks, with the exception of dealing specifically with automotive nodes. This specification is necessary due to the high mobility category of its nodes. Autonomous vehicles are applications within these networks, with highways and smart cities as scenarios. However, applications whose nodes deal with human resources are also suitable.

In localization systems supported by the IoT that use Global Positioning Systems (GPS) or any other global navigation satellite system (GNSS), it is possible to position devices with satellite connection around the globe. However, due to signal loss caused by architectural structures, these systems lack sensitivity when applied to domestic locations. In these contexts, they are called indoor positioning systems (IPS). Even though the definition for IPS can be misleading, since areas such as airports and smart homes are considered indoor, the environments can be rooms or subrooms.

In short, indoor positioning systems seek to locate objects in interior (local) areas such as equipment in smart hospitals (ADARSH et al., 2014), users in smart museums (SPACHOS; PLATANIOTIS, 2020), and even cars in smart parking lots (SYAFRUDIN et al., 2018). Because these applications approach near-real response times, they require constant support from the network. Furthermore, within these scenarios, multiple devices can connect and disconnect from the network at the same time. In these circumstances, the concept of mobility becomes fundamental.

With the expectation to grow from USD$ 3.19 billion in 2018 to USD$ 8.79 billion by 2023 (JACHIMCZYK, 2019). There is an effort from the service providers to prevent wrongful billings for Internet of Things (IoT) applications. Solutions are based on investigating better ways to follow this evolution by supplying Quality of Service (QoS) according to Service Level Agreements (SLA). However, as appointed by (ASLANPOUR; GILL, S. S.; TOOSI, 2020) a selection of incorrect metrics can mislead the developer when it comes to real performance, causing SLA flaws.

The literature on IPS focuses on specific problems requiring a specific number of objects to be located while experiencing limited mobility. By increasing the number of these objects in the localizable area, there is an increase in network latency as well as in server processing, a scenario that is briefly explored, demonstrating a possible gap in the scope of scalability.

When it comes to the domain of the Positioning Systems, migration and service allocation algorithms are based on the OPS. And, because these systems' context-agnosticism is widely encouraged (FIROUZI et al., 2020), they can carry the particularities of these systems to IPS in a context-less form, reducing performance and making them unsuitable for use in all indoor contexts. Some of the wide range of variables of OPS are: user mobility patterns, number and range of fog nodes, starting points, number and duration of user interactions. This way, a study is necessary to understand the behavioral compatibility of both systems, if any.

Context-awareness refers to the property of an entity being informed of its surroundings and is envisaged to achieve intelligent decision making utilizing information gathered through local sources (FAROOQUI et al., 2022). In scenarios of locating devices such as smart buildings and smart hospitals there is no need for dynamic IPS application scenarios that have high turnover and number of devices to be located connecting, disconnecting and reconnecting (GUO et al., 2019) (SPACHOS; PLATANIOTIS, 2020) (BAZO et al., 2021). Based on this principle, distributed systems have the ability to scale, that is, increase their computational capacity by expanding their resources to support the growing workload (LEHRIG; EIKERLING; BECKER, 2015).

Scalability issues in FC architectures for IPS can have significant implications for system maintainability. Furthermore, while brief studies on the performance and mobility of FC-based architectures in indoor contexts have been conducted, a more comprehensive study is required to help validate potential and next solutions. Thus, the interrogations that guide this work are: **can location services in IPS be space-application-agnostic? Are standard fog location service placements from OPS to IPS equivalent?**

1.2 MOTIVATION

Fog computing can reduce network latency by allowing tasks to be executed closer to real time (CISCO, 2015).In this way, the scalability problem in IPS can be better deepened with the investigation of mobility variables in a fog environment. With this research, it may be possible to compare the performance of various architectures while accounting for the constants of each scenario. Furthermore, when it comes to edge device mobility, these agents can have an impact on the performance of the IoT ecosystem, not only through location systems but also through mobility-driven migration architectures. Therefore, the architectures of localization systems must be prepared to maintain a desirable performance.

It is worth noting that the context configuration, which encompasses: the number and positioning of nodes, the service allocation approach, and the very movement of users in *indoor* systems, is also a factor to be considered. The investigation of factors related to the mobility of users, as well as geographic space, allows or does not allow for the existence of proposals for generic architectures. It is defined here as space-agnostic architectures, which are architectures that are independent of their geographic spaces, and application-agnostic architectures, which are architectures that are independent of the type of application being implemented, whether it is a museum or a smart parking lot.

Furthermore, there is encouragement in the IoT and FC communities, such as LF Edge from the Linux Foundation and the Alliance for Internet of Things Innovation, to develop solutions that are application agnostic in order to build an aligned view of the edge and fog computing paradigms (ANTONINI; VECCHIO; ANTONELLI, 2019). The advancement of research can also aid in the advancement of FC-supported IPS. It is emphasized that, although energy efficiency and security are equally important topics, the present study seeks to meet demands based mainly on performance metrics.

1.3 OBJECTIVES

The main objective of this work is **to determine whether standard cloud service locations work equally well on fog-based IPS or at a similar level**. In this way, the following specific objectives were achieved:

- Analysis of open source virtual environments for fog computing that enable indoor positioning systems mobility
- Identification of performance metrics for *fog computing* applied to *indoor positioning systems*
- Study and comparison of *indoor* and *outdoor* environments in the contexts of *positioning systems*
- Research, implementation, and comparative analysis of different contexts and approaches based on fog computing supported by simulation

1.4 CONTRIBUTIONS

As methodological contributions, this work enabled a set of service placement strategies for fog architectures compatible with *indoor positioning system contexts*. In this manner, it was possible to enhance the study also at the edge of the network. When it came to mobility models, a randomized mobility model has also been improved to support mobile nodes in indoor positioning systems.

The adaptation of *outdoor positioning systems* to *indoor positioning systems* through a simulated environment was also made available. The reference models can lay the groundwork to validate derived models that use *fog computing* and *indoor positioning systems*, based on the provided evaluation procedures, such as performance metrics comparison values.

Through the benchmarks developed, it is possible to investigate the bottlenecks when it comes to migration in IPS environments. This concept is relevant for context-awareness issues, since the achievement of intelligent transportation systems could depend on the sharing of contexts among machines. Considering standardized fog nodes location as gateways, it is possible to apply machine learning or stochastic algorithms to evaluate the performance of overall systems that could provide more contextful scenarios.

1.5 SCOPE DELIMITATION

Although energy efficiency and security are equally important issues, this work sought to meet demands based mainly on performance metrics and scalability. It should be noted that the tests presented are specifically focused on performance without packet loss. Besides, as this work investigated the concept of space and application agnosticism, the environments presented tested the corner of the definition of context-lessness by using generic indoor environments and not a specific context.

Due to the distributed nature of fog computing, each node can have different memory and storage capacity. Depending on the scenario, these IoT contexts could be understood as Jungle Computing. To avoid these scenarios, each fog and edge node is uniform when it comes to properties such as downlink, uplink, power consumption, processing power and etc. And since the experiments were standardized, it was chosen the two most traditional service placement approaches. The choice of the placements standing out in this work, is because each represents the conventional extreme for placement: cluster is horizontal placement and edge vertical.

For the edge, besides the computing power being uniform, there are particularities when it comes to mobility. The mobility models used for this study omitted architectonic boundaries such as walls, doors, etc. This occurred because there was no simulation support for loss signals in WLAN environments (indoor or not) at the edge in the tested softwares, which makes the total indoor environment simulation idealistic and therefore, was kept out of the scope.

1.6 DOCUMENT OUTLINE

The research background and state of the art are presented in Chapter 2, and a bibliographic review, as well as related works, are presented in Chapter 3. Chapter 4 provides the research methodology as well as the methodological procedures. Next, in Chapter 5, details regarding the experiments are supplied. Chapter 6 discusses and evaluates the experimental results, and Chapter 7 concludes this study with conclusions and future research.

2 BACKGROUND

Indoor positioning systems, when compared to the outdoors, can seem to be trivial as a subject of study, however these systems are generally adapted from the outdoor perspective and need to be better analyzed, since data mined from environment surroundings can provide the efficient management of them as application and as its applicability. Some of the scenarios for the IPS, for example, is a building of a university campus. The versatility of these systems is rooted in the capacity of using mobility as input and output. Using mobility as input, the system can coordinate services in runtime and as output, can suggest the locality of beacons by mining user data utilizing time as parameter, for example. However, computer paradigms and architectural designs that can truly support these systems is still a discussion, as each of them have their own particularities when it comes to their positive and negative aspects. Therefore, this section presents the state of art of positioning systems, computer paradigms, mobility and strategies for allocating services based on mobility concepts regarding to the scenarios simulated.

2.1 POSITIONING VS LOCATION VS NAVIGATION VS TRACKING SYSTEMS

These systems are made up of two parts: fixed reference nodes (RNs), also known as anchors with known positions, and mobile nodes (MNs), also known as beacons or tags, which are the targets to be located. The location of the MNs is determined by distance, angle from the RNs, or deviation from themselves, (FARAHSARI et al., 2022). Even though these systems can be used interchangeably (SOKOLOVA, n.d.), each of them has its own characteristics. Localization systems find the location related to reference objects, meaning that information regarding the environment is also provided (FARAHSARI et al., 2022). A location system might be attributed to a context with a qualitative approach.

Positioning systems, express the quantitative coordinates of an object in given space, which can be either indoors or outdoors. Inside positioning systems, there are two categories: navigation and tracking systems (Figure 1). Navigation systems actively determine the position and course of an object in relation to a destination point. Tracking systems, as opposed to navigation systems, are used to passively indicate the position of an object.

Mobility involves the position, direction, and velocity of a given device, and depending on the context, like indoor positioning systems, it can also be limited to architectural structures. As a user moves between different access points, the IoT application must deal with the change of location with minimal impact on performance and therefore on the user experience. As a result of the nature of these systems, any of them can be included in location-based systems (LBS), particularly for low-range applications

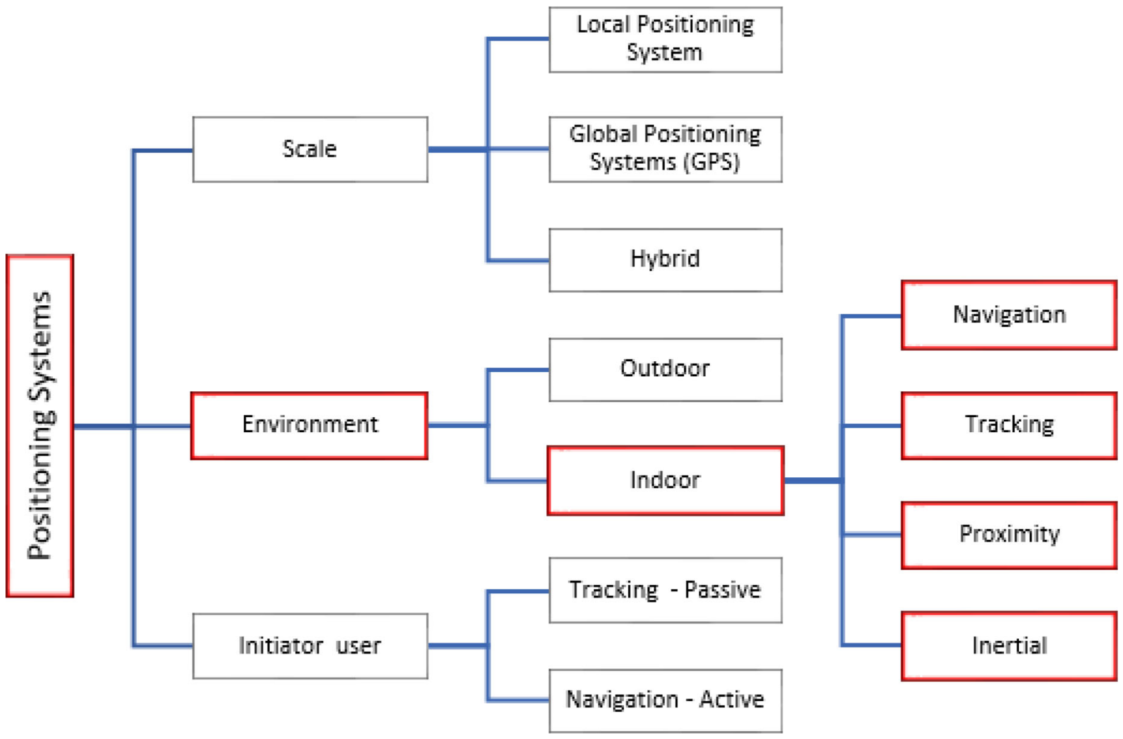


Figure 1 – Positioning system classification from (FARAHSARI et al., 2022)

(SOKOLOVA, n.d.).

2.2 INDOOR POSITIONING SYSTEMS

The rise of IoT and residential and commercial automation has increased demand for location services, including those in indoor environments (MACKEY; SPACHOS, 2017). Popularity can be noted in home care (MARIN; BOCICOR; MOLNAR, 2020) environments, (PIETRABISSA et al., 2013) hospitals, (SPACHOS; PLATANIOTIS, 2020) museums and airports (KARWA, 2019). As indoor systems usually take place in the previously presented environments, there is a lack of definitions for what the maximum range/area is for an indoor environment. In literature, for example, (MAINETTI; PATRONO; SERGI, 2014), categorize indoor environments as rooms and subrooms without defining how large these spaces in meters can be, for example. In this document, **contexts** are referred to as these environments, whether for hospitals, smart homes, or industries.

Positioning systems (PS) are divided into two types: external environments (outdoor) and internal environments (indoor) (MAINETTI; PATRONO; SERGI, 2014). In outdoor positioning systems (OPS), satellite navigation systems are generally used, such as the *Global Positioning System* (GPS). In indoor systems, due to the low power of satellites caused by the loss of signal from the architectural structures, such as the concrete of the buildings, their use is not suitable. Having this particularity in mind, there is a set of possible technologies used for data transmission and reception. Signals can be classified into three types: radio frequency, light and optic, and sound.

Sound-based positioning systems can be ultrasonic or acoustic. Acoustic technologies estimate the position of the objects based on sounds captured. An example of these acoustic systems are the autonomous underwater vehicles. The ultrasonic positioning systems rely on the rate between transmitter and receiver based on the sound velocity to estimate the distance of multiple nodes. SONAR, or sound navigation and ranging, is a navigation technique that uses sound propagation to track nodes in submerged environments.

Visual (or optical) positioning systems, with the help of computer vision and various ML algorithms, can be applied to autonomous driving vehicles, both actively and passively, as well as for locating personnel or things in indoor environments. These systems can be applied to urban areas and underwater environments. Strategies for locating nodes, usually use a camera or a set of cameras. Large databases are necessary for storing features, and depending on the application, advanced processors are necessary when deploying these systems (ANUP; GOEL, A.; PADMANABHAN, 2017).

Radio frequency-based positioning systems can be developed using short-range technologies such as WiFi, Bluetooth (BT), Radio Frequency Identification (RFID), Ultra-wide band (UWB), Zigbee (ZB). Even though all of these technologies are highly available in the market, there are some particularities (BAZO et al., 2021) (SANTOS; AVANÇO; PEREIRA, 2020). BT, for example, covers the range from 2.4GHz to 2.583GHz and is used to locate devices dynamically, enabling real time applications with low energy consumption during transmission and reception of signals. With the 4.0 version, also known as Bluetooth Low Energy or even Bluetooth Smart, device communication no longer requires pairing, and device discovery can be passive and continuous between devices and applications.

The Zigbee protocol operates at a global frequency range of 2.4GHz. Despite the fact that it can cover a greater distance than BT, this protocol is intended for low-data-transmission applications. It is often chosen due to its low cost, energy efficiency, and off-the-shelf components, which reduce hardware complexity. It is widely used by the home automation and care industries.

RFID is a ubiquitous technology that can be defined as a system containing electronic circuits attached to things. It is necessary to have three components: a tag attached to the thing, the reader (device responsible for detecting the presence of the target tag), and the antenna, the passive device that is used by the reader to transmit the radio frequency signals. The active tag contains a battery and can transmit signals autonomously, whereas passive tags have no battery and require an external source to emit signals.

UWB provides high precision, low energy consumption, small tag size, and operates on frequencies ranging from 3.1 GHz to 10.6 GHz, with the ability to handle

Table 1 – Comparison of wireless radio frequency IPS technologies

	Accur. (m)	Cov. (m)	Cost	Compl.	Energ. C.	Env.
WiFi	10-800	20-50	Medium/Low	Low	High	Indoor/Outdoor
BT	1-10	1-30	Low	Low	High	Indoor/Outdoor
ZB	1-200	1-30	Low	Low	High	Indoor/Outdoor
RFID	0.5-5	1-10	Low	Low	Low	Indoor
UWB	1-15	High	Medium/Low	Low	High	Indoor/Outdoor

interference and provide high precision tracking down to 20 centimeters. However, its widespread usage still depends on greater market availability and lower cost. Using wide bandwidth, which can be defined as for IPS and OPS, this technology is used to transmit data.

The Table 1 presents a comparison of the IPS technologies based on their capabilities: accuracy, coverage, cost, energy consumption, and environment. WiFi, Bluetooth, Zigbee, RFID and UWB represent the most used technologies for wireless solutions.

2.2.1 Wireless Local Area Network IPS

Wireless Fidelity, or as its acronym is known, Wi-Fi, is the most well-known communication technology, It is based on the IEEE 802.11 standard and operates in the ISM radio band at frequencies of 2.4 and 5 GHz, as well as 1—6 GHz in its 802.11.ax version (MAINETTI; PATRONO; SERGI, 2014). When applied to a local area network, Wi-Fi, can be used to estimate the location of a MN within the network. Even though Wi-Fi is not suitable for localizing small objects, it is convenient and accurate enough for tracking human beings in IoT applications.

One of its main advantages is its cost effectiveness, since nowadays the majority of devices are equipped with this technology without the need of installing new software, besides its high data rate and availability in most devices (MAINETTI; PATRONO; SERGI, 2014). In these networks, a node emits or receives signals from or to the wireless router, which determine the precise location of the target device. One of the remarks regarding the WLAN IPS, is the fact that these systems heavily depend on the placement of beacons, since the quantity of the beacons can lower the system's performance and accuracy and increase the total cost of the system (ANUP; GOEL, A.; PADMANABHAN, 2017).

The algorithms (or techniques) for locating devices can be characterized into three groups: time based, direction-angle-based, and signal-strength-based. Time-based techniques use the time of signal propagation to define the mobile node's location, which can be based on the propagation time between receiver and transmitter without the need for time synchronization (Time of Arrival), the difference between arrival times in multiple anchor nodes (Time Difference of Arrival), or even the time it

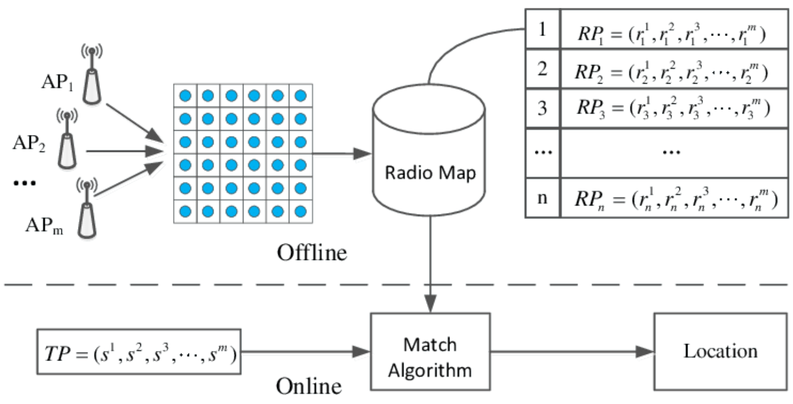

AP$_1$
AP$_2$
...
AP$_m$
Radio Map
Offline
1 $RP_1 = (r_1^1, r_1^2, r_1^3, \cdots, r_1^m)$
2 $RP_2 = (r_2^1, r_2^2, r_2^3, \cdots, r_2^m)$
3 $RP_3 = (r_3^1, r_3^2, r_3^3, \cdots, r_3^m)$
n $RP_n = (r_n^1, r_n^2, r_n^3, \cdots, r_n^m)$
$TP = (s^1, s^2, s^3, \cdots, s^m)$
Online
Match Algorithm
Location

Figure 2 – Wifi fingerprint based on (ZHAO et al., 2018)

takes for a signal to be transferred from transmitter to receiver and back (Round Trip Time).Direction based techniques rely on the angle of the received signals, requiring anchors to determine the intersection of multiple orientations using directional antennas (Angle of Arrival). The power and strength of the received signals are used by signal-based systems to determine the MN's location.

In WLAN contexts, it is generally used as the Received Signal Strength Indicator (RSSI) because of its accessibility and popularity. This method is based on determining the received signal strength values (VARSHNEY; GOEL, R. K.; QADEER, 2016).RSS can be used to locate a device using the following techniques: angulation, lateration, and fingerprinting.Lateration is a method that determines the position of an unknown object based on the distance from more than two known reference nodes. Trilateration, which is the most popular use of lateration, uses three reference nodes. Angulation utilizes the geometric measurement of the arrival angle of the signal, using Angle of Arrival techniques.

For Scene Analysis and Pattern Matching, spatial information of specific points is collected to form a database. Fingerprinting (Figure 2 is a method using RSSI that is classified in the given approach. In this technique, there are three phases: area mapping and storing, comparing measured RSSI values to predefined values in the database, and then mapping the current position. Even though this approach may result in accurate positions, remapping is needed for any change.

2.3 ENABLING IOT COMPUTING PARADIGMS OR IOT-ENABLED COMPUTING PARADIGMS

Cloud computing has, for a long time, been the most widely used and stable technology for IoT applications. However, nowadays, due to the increase in possible applications, it has become necessary to develop new, singularity-driven IoT Computing Paradigms. The Figure 3 shows a summarized range of these current IoT paradigms. As the intersection of all the paradigms, the main goal is to bring computing resources

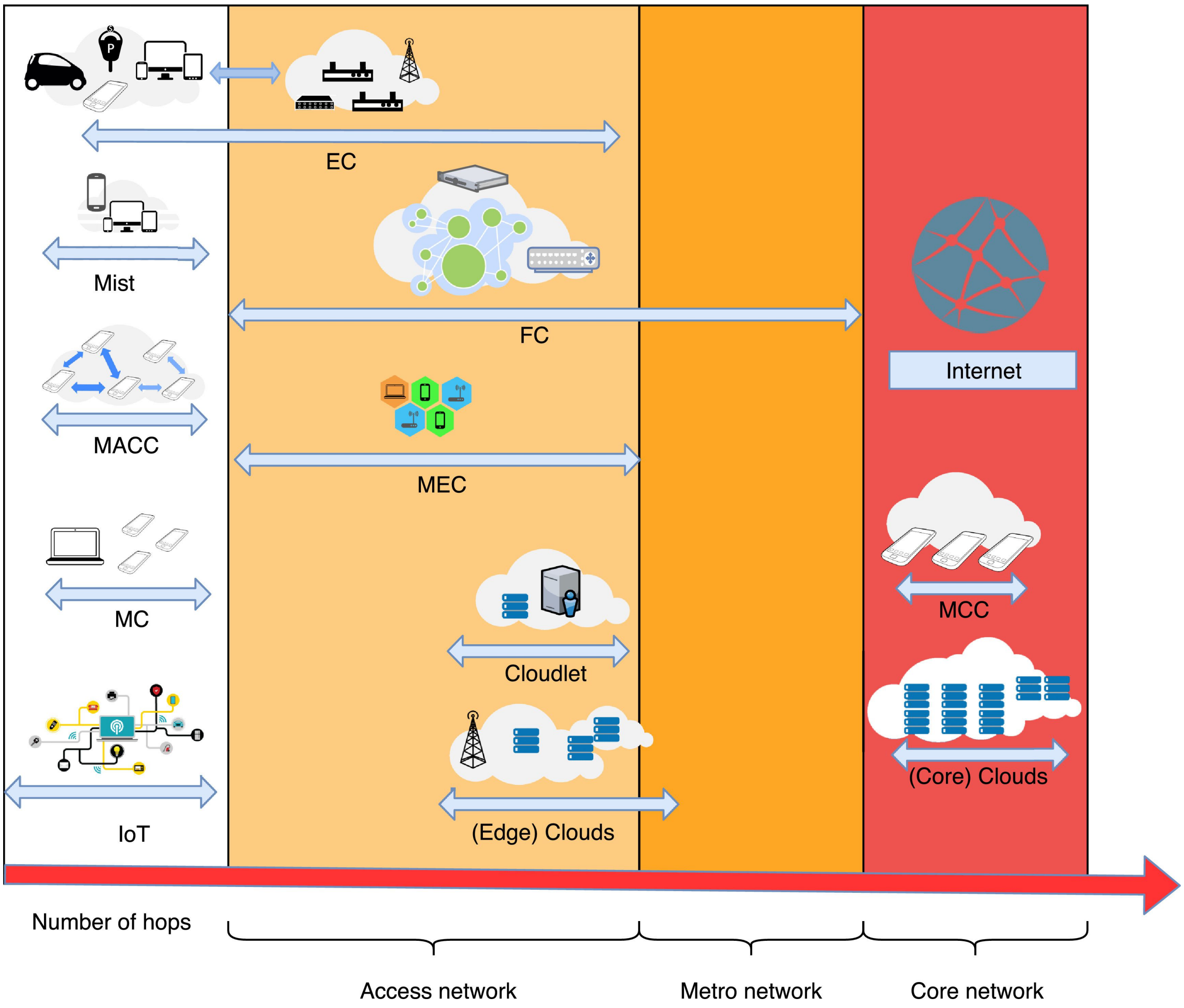


Figure 3 – IoT Computing Paradigms (YOUSEFPOUR et al., 2019)

closer to the users, reducing latency, costs and increasing QoS; however, each of them differs on the technologies and applications. As this computing paradigm can be defined as a set of computing resources geographically far from the devices with high computation power and availability, devices can only connect to the cloud through the core network.

Mobile computing (MC), also known as nomadic computing, has as its agents portable devices, such as laptops, tables, and phones. Used to create pervasive context aware apps, such as location-based ones. As aspects, these carry, low processing power and require low or no hardware implementation since they are connected through BT, Wifi, ZigBee or others. They are networks with limited resources that employ a distributed architecture.

Edge computing (EC) improves the management, storage, and processing of data generated by MCC; however, unlike this computing paradigm, the edge is not located on IoT devices, but rather at the network's edge close to them. The Open

Edge Computing Initiative (YOUSEFPOUR et al., 2019) defines it as the processing or computation done at the edge of the network through small data centers. Through ubiquitous concepts, common uses are filters, pre-processing intelligently using specific cloud services closer to the devices. EC Because the devices are interconnected and decentralized, they have lower latency when compared to the MCC and CC, as well as higher service availability.

Mobile cloud computing (MCC) is a hybrid of cloud and mobile computing. This paradigm processes and stores data outside of the mobile device, as apps are not just destined for the devices. It can be used for crowd sourcing, healthcare, and sensor data processing, and it can use dynamic offloading for computationally intensive apps. However, as it aims to connect cloud providers and mobile devices, challenges regarding connectivity are an open issue. It requires large-scale data centers.

Mobile ad hoc networks (MACC), which use temporary and dynamic topologies to accommodate in highly unstable environments, may not be a suitable option for poor infrastructure. It can be used in disaster relief, video streaming, and autonomous vehicles by forming small clouds, similar to clusters. Mobile devices perform the functions of data providers, storage devices, and processors. These devices are also in charge of routing traffic among themselves and are computationally intensive. Services are available through mobile devices connected through radio frequency technologies, such as WiFi, BT and other cellular protocols.

MEC extends to other radio access technologies, such as 5G and 4G, within the Radio Access Network, closer to mobile devices. It can be used in video analytics, connected vehicles, health monitoring, and augmented reality. It uses virtualization, can use WAN, cellular connections, has a focus on network infrastructure providers and mobile devices, can use SDN, NFV, and can deliver mission-critical, delay-sensitive apps over the mobile network.

Cloudlets are typically used in WiFi environments to bring the cloud closer to users by utilizing resource-rich computers that can be organized as clusters.They use miniaturized clouds that are only a one-way hop away from the devices. The aim is to offload computation from devices to dedicated VMs. high potential for wearable assistance. can support local services for devices that distribute tasks among cloudlet nodes in close proximity via WiFi, Bluetooth, and Bluetooth Low Energy.Devices can be providers or clients.

The Mist computing paradigm refers to processing at the extreme edge, like in a wearable, mobile device, smart watch, or smart fridge. It can be considered a subset of MACC. It is used when there is a necessity to reduce the load of WiFi infrastructure. It is worth noting that the devices can function as thin clients or servers. Since this paradigm is self contained, it can preserve the privacy of users data using the local processing.

2.4 FOG COMPUTING

The concept of fog computing (FC) is introduced so as to enhance IoT systems scalability, reactivity, and efficiency using moderate computing resources at a low power level. Although there may be exceptions, the hardware can be located closer to the clients.Fog can be accessed through connected devices from the edge to the network core. The fog nodes do not necessarily need to be connected or even have a fast Internet connection, thus providing offline or even real time apps.

This paradigm aims to extend the Cloud Computing paradigm to the edge of the network, allowing a variety of "silent services" (NASIR et al., 2019). Through high virtualization, it provides compute, storage and networking services between edge devices and *data centers* of *Cloud* (TRAN et al., 2020). It causes decentralization of the flow of applications in distributed services and, as a result, reduced latency and energy costs (MANSOURI; BABAR, 2021) by allocating processing power closer to the sensors and actuators.

As they are closer to the devices, they are used when there is a need for pre-processing and monitoring of short-term data and in critical real-time applications (CISCO, 2015). Another feature is that FC supports large-scale sensor networks (BONOMI et al., 2012).This property has been used in the context of real-time systems (BATTISTONI; SEBILLO; VITIELLO, 2019) or even for low latency purposes (GUO et al., 2019). However, as a technology, formal architectures are rare and not accepted as standardized for IoT applications (NASIR et al., 2019).

2.4.1 Fog computing vs. Edge computing

Although many authors assign *Edge* as a synonym for *Fog*, both paradigms carry particularities (IORGA et al., 2018). The OpenFog Consortium defines the fog as having hierarchies, thus providing computing, networking, storage and control from the cloud to the things. The edge is limited to the computing from the end devices (sensors and actuators).

- *Fog computing*: runs on applications in a multi-layered architecture that allows dynamic reconfigurations for different applications while performing intelligent computing and streaming services. It is hierarchical and has support for storing, controlling, and accelerating data processing.

- *Edge Computing*: runs specific applications in a fixed logic location and provides direct service transmission. It is usually limited to a few peripheral devices.Depending on the architecture and context, this could be referred to as the IoT layer.

In this way, FC can be seen as a transformation of data into information, in which at each layer the data analysis decreases the amount of transmitted data and increases the "quality" of the information (BATTISTONI; SEBILLO; VITIELLO, 2019).

2.4.2 Standard fog architecture

Different researchers may use different models; however, in the most common FC scenarios, three main layers are used: *Cloud*, *Fog* and *End Devices* (MARGARITI; DIMAKOPOULOS; TSOUMANIS, 2020):

- *End Devices* or *IoT layer*: lower layer that represents sensors, actuators, mobile devices, etc., has computational capabilities and forms a communication network whose data is transmitted to the Cloud through Fog.
- *Fog layer*: layer above the devices, where any device that can process, store, and connect to the network can be a fog processing device. Considering this, there are devices that can be both IoT and fog. Resources are interconnected, and devices form the distributed system that provides services to a set of end devices at a specific location and manages the data that is transmitted by these.
- *Cloud layer*: layer above the Fog consisting of physical data centers in which each node has a CPU, main memory, and broadband network and is used for user requests for resources. Clouds are connected by WANs and provide intense data analysis to end devices. They provide high-quality services with a high level of fault tolerance.However, they have high latency due to WAN connections.

This work employs architectures as a set of configurations in the three layers, as presented in Figure 4. Models and architectures will be considered synonymous.

2.5 STRATEGIES FOR SERVICE PLACEMENT

Because of the limited coverage of access points and multi-hops, the quality of service may suffer; thus, better and optimal placement, closer to the newest user location, is required in IPS. Aspects like message size, available bandwidth, and load on the target node are also taken into account when choosing the optimal destination node. This process is called migration.

Service migration departs from one node (source) to another (destination). Such service migration depends on the location and identification of the intermediate node to which the migration source can upload the application of the corresponding service and the target migration destination can download it, if there is no direct connection between the respective migration points, even though positioning systems remain an open challenge for mobile applications.

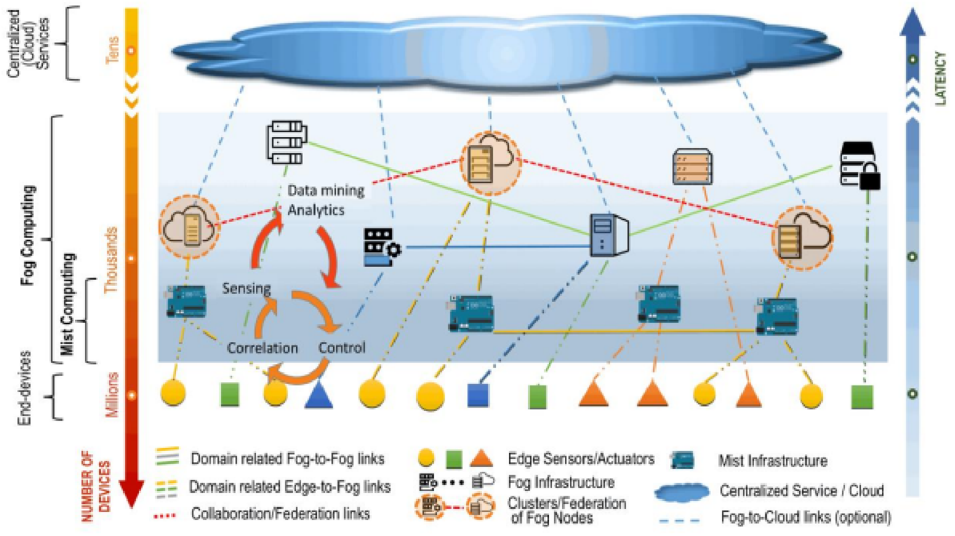

Centralized (Cloud) Services
Fog Computing
Mist Computing
End-devices
Tens
Thousands
Millions
NUMBER OF DEVICES
LATENCY
Data mining Analytics
Sensing
Correlation
Control
Domain related Fog-to-Fog links
Domain related Edge-to-Fog links
Collaboration/Federation links
Edge Sensors/Actuators
Fog Infrastructure
Clusters/Federation of Fog Nodes
Mist Infrastructure
Centralized Service / Cloud
Fog-to-Cloud links (optional)

Figure 4 – Standard fog computing architecture from (IORGA et al., 2018)

To address this issue, different strategies for service placement and migration have been developed, which, depending on the context, can have better or worse performance. The main categories for these strategies are horizontal or vertical placements. The following subsections present an overview of the state of the art of clustering (horizontal) and edgewards (vertical) methodologies, taking into account positioning systems in fog architecture systems.

2.5.1 Clustering placement

In this type of placement, each group has an orchestrator node (proxy) and N worker nodes (gateways). The gateways of each group in the fog are interconnected, as shown in Figure 5. The focus in question, is on the horizontal scalability of the nodes. By decentralizing placement management and assigning responsibility for decision making to each node, service discovery and load balancing become crucial points in this approach.

Aiming for horizontal scalability, when configuring the clustered approach to the architecture, the same migration strategy is also used. In the present case, by connecting all gateway nodes in a mesh network, migration of modules directly between source and destination is allowed. It is noteworthy that nodes in the upper layers are not involved.

In this approach, since the relationships among clusters are blind and have to be managed individually, a drawback regarding migration can be an issue for intercluster events. This event can have a direct impact on overall system performance because if the target cluster is busy, the migration will be delayed because intercluster resources are not shared.

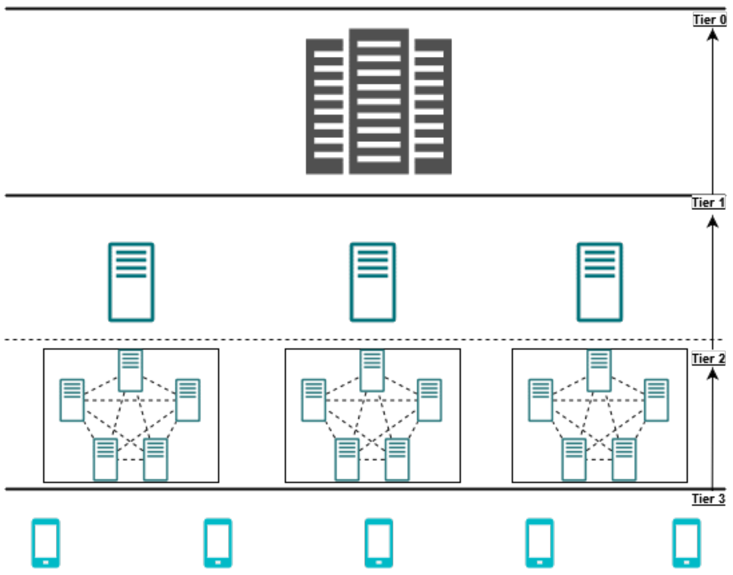


Figure 5 – Topology using clustered placement

2.5.2 Edgewards placement

Based on the First Come First Serve (FCFS) strategy, the edge-ward placement, or edgewards, favors the deployment of modules at the edge. If there are insufficient resources, the approach moves the action to be performed to the upper layer until the action in question is performed, even if it is only in the cloud, as shown in Figure 6.

The edgewards approach has better performance when the architecture of fog devices is hierarchical (SILVA, D. M. A. d. et al., 2019). As a disadvantage, due to the low computational capacity of the edge nodes, there is a tendency to make the application as a whole more expensive; after all, there is greater use of better equipped nodes (cloud data centers) or, depending on the architecture, also layered fog nodes. Figure 7 shows the topology configuration using this placement.

The application flowchart based on the application model with three states is depicted in Figure 6. Raw_data (initial request), filtered_data (pre-processed request), and processed_data (complete request) are the three states.

2.5.3 Clustered and Edge-ward placements in OPS

The authors of (MAHMUD et al., 2022), defined a benchmark for understanding how well these placements (clustered and edgewards) perform. The outdoor environments defined in the EUA dataset in the industrial Melbourne Central District are used as the targeted positioning system in this paper.The experiments show there is a superiority of clustering migration when compared to edgeward and cloud-centric approaches, as described in the Audio Translation Service (ATS) experiment, and also for network usage and coordination among computational resources in Cardiovascular Heath Monitoring (CHM).

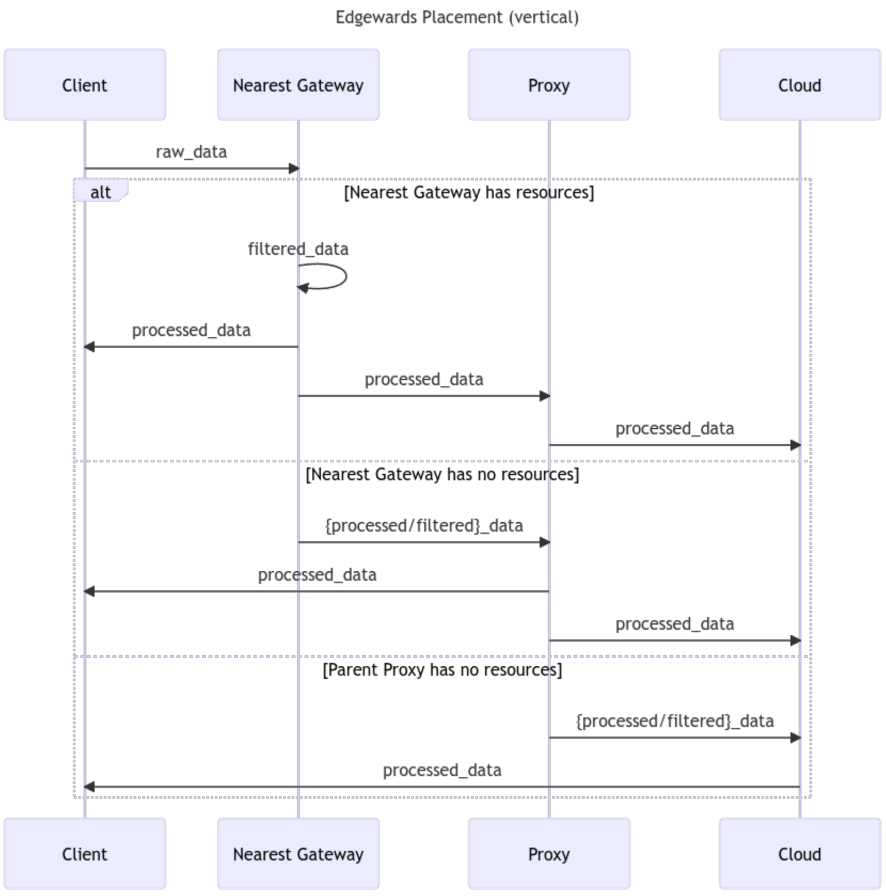

Edgewards Placement (vertical)
Client
Nearest Gateway
Proxy
Cloud
raw_data
alt
[Nearest Gateway has resources]
filtered_data
processed_data
processed_data
processed_data
[Nearest Gateway has no resources]
{processed/filtered}_data
processed_data
processed_data
[Parent Proxy has no resources]
{processed/filtered}_data
processed_data
Client
Nearest Gateway
Proxy
Cloud

Figure 6 – Application flowchart using vertical placement

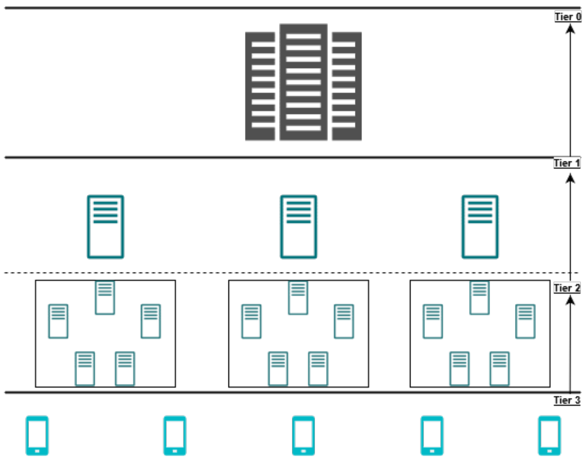


Figure 7 – Topology using vertical placement

2.6 AWARENESS VS AGNOSTICISM

Buzzwords such as context agnostic and application agnostic have been explored in academia (PSOMAKELIS et al., 2020) (PHUNG; YOUNG; ZOMAYA, 2017). In computer science, a software program is said to be agnostic or data agnostic if the method or format of data transmission is irrelevant to the device's or program's function. Devices or programs can receive data in multiple formats or from multiple sources, and still process that data effectively. Therefore, agnosticism, in information technology contexts, refers to the generalization that can represent interoperability among various systems. The term can refer to software and hardware, for example, and some say any type of environment (VANINI; GIORDANO, 2013).

Agnosticism is used to define architecture designs in a variety of fields. Cloud agnostic providers refer to an organization migrating their production workloads to another cloud provider, and language agnosticism in programming refers to the ability to develop efficient software in any language. Besides the inclusiveness of application agnosticism, there are some risks when developing a context agnostic architecture. For instance, the term *space-agnostic*, meaning that given an architecture, it can perform according to the SLA or QoS metrics besides the area (in meters per square). However, different variables must be considered in positioning systems, particularly when it comes to mobility.

In some way, for location-based systems (LBS), the terms "context" and "application" can be used interchangeably. As placement strategy plays an important role in an architecture, a generic one for OPS can appear to be ideal for IPS surroundings, but the change-over of variables can mislead the developer. This way, if a part of an architecture needs to be changed to be accepted, the architecture is not agnostic anymore. Because agnostic OPS approaches do not always cover all aspects of IPS, context awareness should be preferred over agnosticism.

2.7 VIRTUAL EXPERIMENTATION ENVIRONMENTS

The virtual environments mentioned in this project are means of experimenting before they are implemented in reality.This methodology can help to isolate the study from real problems before there is any loss. Within this framework are included: simulators, emulators, and *testbeds* (GILL, M.; SINGH, 2021).

- Simulators are capable of reproducing each step of a system, providing environments that are minimally faithful to reality before the implantation process (SILVA, H. R. S.; COUTINHO, 2018).
- Emulators are software or hardware systems capable of behaving like any other system and, therefore, are able to replace the original system with little loss of performance (SILVA, H. R. S.; COUTINHO, 2018).
- Testbeds are built from prototypes and component parts of a real system and are used to study the elements of a system and therefore focus their study on specific parts of the whole. (FORTIER; MICHEL, 2003).

For fog computing, there are some specialized simulators, such as: iFogSim, MyiFogSim, EdgeCloudSim and YAFS (KUNDE; MANN, 2020). For emulators: Fogbed

(COUTINHO et al., 2018a), Fogify (SYMEONIDES et al., 2020) and EmuFog (MAYER et al., 2017). As testbeds, there are works by *National Chiao Tung University* (HUANG et al., 2017). This document focuses on Open Source Simulators in order to develop and evaluate fog-based architectures.

2.7.1 Indoor Positioning Systems simulators

Within IPS, there are some simulators, such as: SMILe (JANKOWSKI; NIKODEM, 2018), Navindoor (OROYA-VILLALTA; DÍEZ; BAHILLO, 2019) and Pylayers (AMIOT; LAARAIEDH; UGUEN, 2013). As some testbeds exist based on (ADLER et al., 2013) grids and as reference model (SCHMITT et al., 2012). Considering the main simulators used in the literature, it is noted that these systems place a greater focus on the analysis, testing, and evaluation of localization methods. It is important to point out that although it is possible to develop any IPS scenario, none of them provide support for testing networks, whether fog or cloud.

2.7.2 Fog computing simulators supporting mobile nodes

As a computing paradigm becomes better understood and standardized, simulation software is capable of defining its most important characteristics. These systems are capable of defining topologies, designs, architectures and also determining whether it is worth implementing a system in the real world. For fog computing simulators, for example, it is possible to define the computing resources and numbers of fog nodes and devices, as well as tiers and even placement and migration policies for services, in order to determine whether and how an architecture can be improved.

For the variety of fog simulators, not every one has support for mobile nodes in the IoT ecosystem. This is because the concept of supporting mobility in terms of handling specifically migration events in run-time is still new to the fog.In order to evaluate mobile nodes in IoT, it is important that the simulator support this functionality. Open source simulators supporting mobility used nowadays, are iFogSim (MAHMUD et al., 2022), MyiFogSim, EdgeCloudSim and YAFS (KUNDE; MANN, 2020). Throughout the development of this paper, each simulator was analyzed based on two perspectives: research and development.

As for researchers, it is important that the simulator have a solid grounding in the theory. We decide to measure the acceptance in academia using the number of citations through the Google Scholar platform [1]. As the iFogSim simulator study group updated the software to version 2, demonstrating the new functionalities and its background, there was another paper, which we kept separately, as presented in Table 2.

[1] https://scholar.google.com/

Table 2 – Comparison of fog simulators regarding academia aspects

	Year	N° of citations
iFogSim	(GUPTA et al., 2017)	1365
iFogSim v2	(MAHMUD et al., 2022)	32
MyiFogSim	(LOPES et al., 2017)	101
EdgeCloudSim	(SONMEZ; OZGOVDE; ERSOY, 2018)	429
YAFS	(LERA; GUERRERO; JUIZ, 2019)	143

The iFogSim was the first open source fog simulator available worldwide. It was developed by the Cloud Computing and Distributed Systems (CLOUDS) Laboratory, a study group located at the University of Melbourne in Australia. The simulator is an evaluation platform that enables the quantification of the performance of resource management policies on an IoT or Fog computing infrastructure in a repeatable manner (GUPTA et al., 2017). It is possible to measure the impact of resource management techniques using this software in terms of latency, network congestion, energy consumption, and cost. Nowadays, it supports mobility driven events, such as user interactions (movements), in run-time (MAHMUD et al., 2022).

MyiFogSim was developed in the Computer Network Laboratory at the Institute of Computing at Unicamp with collaboration from researchers at Western University. It aims to address resource allocation in fog computing in the face of user mobility, supporting mobility through the migration of virtual machines between cloudlets (LOPES et al., 2017) as it extends the first version of the iFogSim.However, the direction and speed of the presented mobility models lack customization.

EdgeCloudSim is an open source simulator based on the CloudSim simulator that was developed at Bogazici University to address edge computing demands. Different architectures are supported (n-tiers) as well as mobility models (SONMEZ; OZGOVDE; ERSOY, 2018). Despite supporting various networking resource customization, service placement and allocation are not mentioned in the presented article.

Differently from the previously presented simulators, which were built in Java, YAFS uses the Python language. This open source software was developed in the University Of Balearic Islands and provides a wide variety of customizations of architectures, from the allocation of resources, billing management, network design, and so on (LERA; GUERRERO; JUIZ, 2019). Yet, when accessing the online documentation available on the Github platform, aside from the absence of code implementations and theoretical background, there were sections still in development (alpha version).

The most cited simulator was the iFogSim, followed by the EdgeCloudSim, YAFS and finally the MyiFogSim. As for the year of publication, chronologically, it starts with the first version of iFogSim, then, MyiFogSim, EdgeCloudSim, YAFS and ultimately version two of iFogSim. Even though the first version of iFogSim, did not have support for mobile nodes, this project was the only one to publish more than a paper version featuring the new functionalities.

From a developer perspective, as all of the simulators are open source, metadata is publicly available. In these cases, popularity can be measured in Github using the number of stars - because it is used to demonstrate interest citegithubStars - the date of the last update, and the number of open issues in Github. For the last topic, the result was based on the date of February 8th, 2023 as presented in Table 3.

Table 3 – Comparison of fog simulators regarding development aspects

	Last update on	**N° of Stars**	**Open issues**
iFogSim	Sep 20, 2022	78	5
MyiFogSim	Sep 29, 2020	8	3
EdgeCloudSim	Nov 2, 2020	338	37
YAFS	Nov 23, 2022	66	2

For the analysis of the documentation coverage, an important factor had to be taken into account: the comparability of indoor to outdoor environments should be possible. YAFS comes closer to this possibility but does not support the latitude and longitude (locality) of fog nodes or either of the edge devices. Only iFogSim v2 had geographical support for the creation of the environments, providing a more suitable configuration for comparing IPS environments to OPS ones. However, there is a lack of documentation for placement algorithms, which makes creating new ones difficult. Finally, it is worth noting that the iFogSim supports applications in real-time (AWAISI et al., 2021).

2.8 MOBILITY MODELS

A mobility model represents the moving behavior of each mobile node. (ARIYAKHAJORN; WANNAWILAI; SATHITWIRIYAWONG, 2006). These models can be defined by two types: synthetic and mobility-trace-based base models. Synthetic base models can be generated using statistical data in such a way there is a pattern. These models are known to have low deployment costs and easy implementation. Nonetheless, there is little resemblance to reality, which adds to the study's complexity.On the other hand, mobility traces can mimic real life mobility. These models can show more simple patterns through large datasets and, therefore, high overhead (BATABYAL; BHAUMIK, 2015).

The challenge for developing and using mobility models is that they should be as close as possible to reality. For that, there are some categories of mobility models used in MANETs that are accepted and used in the literature of the positioning systems. The authors of (BAI; HELMY, 2004) use as main categories the random models and the ones with temporal, spatial, and geographic dependency. Figure 8 shows the organization developed by the authors.

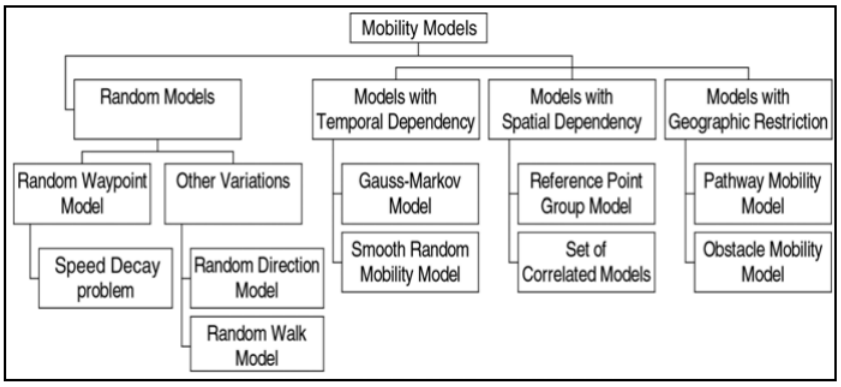


Figure 8 – Categories of mobility models used ind MANETs from (BAI; HELMY, 2004)

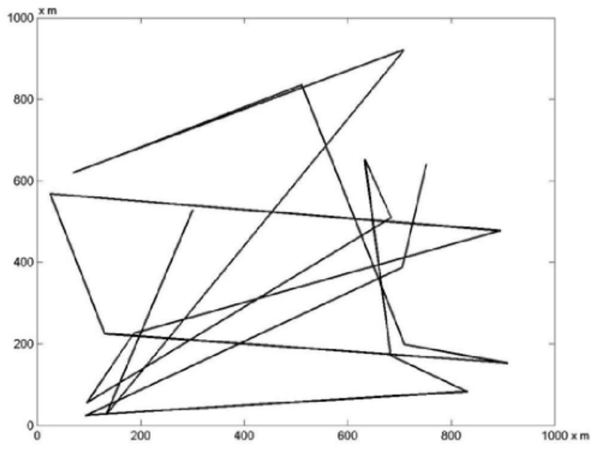


Figure 9 – Traveling pattern of an MN using the Random Waypoint mobility model from (ARIYAKHAJORN; WANNAWILAI; SATHITWIRIYAWONG, 2006)

2.8.1 Random waypoint

Random Waypoint Mobility Models (RW) are based on the direction and/or speed value changes, including the stopping of movements for a certain period of time (CAMP; BOLENG; DAVIES, 2002). Inspired mobility models use geographic or geometric boundaries as a threshold so that the MN does not exceed the simulation area and can turn around for the next interactions.Because of its simplicity, it is widely used in simulation research, and is included by default in these softwares.However, this implementation, like in the iFogSim v2, is prepared only for outdoor positioning systems (MAHMUD et al., 2022). We were able to convert this mobility model to geographical (latitude and longitude) indoor positioning systems as presented in the Appendix A.

Differently from the Random walk and Random Direction, the RW mobility pattern involves two states of interaction: walking and stopping (pausing). This alternation of states can provide a more accurate simulation when it comes to user movements in IoT dynamic environments. Figure 9 depicts a traveling pattern using RW.

3 BIBLIOGRAPHIC REVIEW

In this section, works related to the theme of this document are presented through a bibliographic review. Three articles were selected as relevant to this research. At the end of the section, we provide a discussion regarding the proposals of each article and ours.

3.1 RESEARCH PLANNING

In order to define the strategies to be adopted, the objective is to understand ***the current scenario of IoT research supported by the Fog Computing paradigm in wireless IPS contexts***. To assist in the research, 4 specific questions were defined, listed below:

- QE1: Which localization technologies and methods are being studied?
- QE2: What are the context(s) for applied IPS?
- QE3: How important is scalability?
- QE4: What are the metrics used to evaluate the work?

The databases ACM DL, IEEExplore, Science Direct, Springer, Web of Science and Scopus were chosen as search sources. The Table 4 displays the references of the search elements used for better visualization.

Table 4 – Synthesis of search elements

Acronym	Search element
IoT	*("IoT" OR "internet of things")*
FC	*"fog computing"*
In	*"indoor"*
Lc	*("location" OR "localization" OR "locationing" OR "navigation" OR "tracking" OR "position" OR "positioning")*

The search expression was applied to the abstracts of the works in order to find the most relevant searches for this section; more details will be provided in Phase 1.The number of results returned from each of the sources for the expression used is shown in Table 5

Table 5 – Relation between the search expression and the results obtained from the sources

Expression	IEEE	ACM DL	Springer	Web of Science	Scopus	Science Direct
IoT + FC + In + Lc	3	1	102	0	12	1

Table 6 – Criteria used in the phases

Exclusion Criteria
E1 - Not written in English
E2 - Be published after the 2016-2021 time frame
E3 - Not having the content of the search *string* in the summary
E4 - Not being a complete article
E5 - Being a survey, a systematic literature review, or mapping without results
E6 - Do not use wireless location technologies

3.2 PRIMARY STUDIES SELECTION

With the criteria defined above, the following steps were defined for the selection of *papers*:

- Phase 0: Duplicate Removal
- Phase 1: Reading the abstract of each article and applying the criteria

3.2.1 Phase 0

Due to the selection of multiple sources, it becomes necessary to reduce the chances of reading duplicate articles. This step was performed by comparing the titles of the papers, making it possible to reduce from 119 papers to 116.

3.2.2 Phase 1

It is important to note some nuances of the fonts:

- The Springer source does not have a search feature in the abstract, being necessary to read each one. The feature for including preview content in searches has been disabled.
- For the Web of Science, the Capes Portal was used with search by subject.
- The Science Direct font searches for both title, keywords, abstract and highlights.
- For the Science Direct source, as it is only possible up to eight Boolean combinations, it was necessary to divide the search element Lc, described in the Table 4, into two parts: one with just the term *("navigation" OR "tracking" OR "position" OR "positioning")* and another with the term *("location" OR "localization" OR "locationing")* in its place, returning the same work and accounting for 1 .

Criteria E1 and E2 were applied first because they could be combined with the use of search expressions, which was made possible by the support provided by the source search engines.Afterwards, criteria E3 and E4 were applied, excluding all articles that were also conference presentations, leaving six articles. Next, criteria E5 and E6 were performed in order to investigate only *papers* applied with results.

Although the search expression used for this review is "IoT + FC + In + Lc", out of curiosity, the word *"scalability"* was added to check which papers mention scalability in the abstract. The only sources that returned results were Springer with 62 and Science Direct with 1 result.

- Springer: In the case of Springer, as the source does not support searching within the abstract, all the abstracts of the indicated papers were read, and there were no papers that met the expression criteria in the abstract.
- Science Direct: In the case of the Science Direct source, it was possible to identify a work that met the expression requirements, but because it is a literature review, and the term *"indoor"* appears only in *Highlights* it was not possible to consider her either.

The results of the steps were organized in a spreadsheet available in the footnote [1].

The quality assessment stage was skipped due to the work-results' suitability to the required scope.

3.3 RELATED WORKS

In (BATTISTONI; SEBILLO; VITIELLO, 2019), published in 2019, the objective was to develop a *framework* of an *application agnostic* architecture using low-cost components with an internal navigation system as a use case. In this article, the authors consider *Edge* as being the execution of specific applications in fixed locations connected directly with Cloud services. In this way, they mention another component in their architecture: *Mist*.

Mist Computing is a more specialized paradigm with smaller computational resources and are located even closer to the end (user) devices. In the *paper*, this layer is supported by the multi-hop network. *Mist* nodes generally share the same location as end devices, to which they provide low-latency computational services.

The authors use ESP32-WROVER as a Mist node, Raspberry PI 3 Model B+ as a Fog node, the MQTT protocol and the MQTT Mosquitto *broker*. The MQTT protocol is used as a notification system for the proposed navigation and position systems. Because it is a low-cost proposal, the authors use *beacons Bluetooth Low Energy* (BLE).

[1] Worksheet for the Literature Review: `https://docs.google.com/spreadsheets/d/1oP4S_OrUwbY4LKM2glS9iMettqO6E7DwGMeJhWYu-zQ/edit?usp=sharing`

In the work of (PEŠIĆ et al., 2018) a real-time positioning and estimation system based on *beacons* BLE is proposed. The system can detect signal propagation from obstacles, enabling disturbance correction, trajectory exploration, and self-discovery via machine learning. Location estimation is semi-supervised and highly accurate.

BLEMAT, the authors' proposal, is sensitive to context, that is, changes in the environment such as: different flows of people over a period of time or new obstacles. Although the proposal is based on three levels: *sensor, fog and cloud*, the authors' focus is on testing, application, and evaluation of algorithms. In this architecture, the Fog level performs trilateration based on measured RSSI values and is responsible for reactive decision-making, such as Kalman filter application and corrections.

In (PEŠIĆ et al., 2019), the authors continue with (PEŠIĆ et al., 2018), applying the BLEMAT system in a proposal for managing *smart buildings* with data analysis and neural networks. The test environment is a residential building with Wi-Fi access points acting as BLE *beacons* and providing Internet access.The *dataset* is available online.

3.4 RESULT ANALYSIS

In a general analysis, it can be noticed that two of the three works are related and follow-ups by the same group of researchers. Concerning the answers to the previously set research questions, they are presented below:

QE1. All studies used Bluetooth location technology and *Received Signal Strength Indication* as the location method. In (PEŠIĆ et al., 2018) and (PEŠIĆ et al., 2019), machine learning (ML) and a Kalman filter are used to improve localization and estimation. All items also use mesh netting.

QE2. In (BATTISTONI; SEBILLO; VITIELLO, 2019), the authors study a generic use case (office with several branches), while (PEŠIĆ et al., 2018) experiments with offices with applications in *smart buildings* with a model declared *semi -space-agnostic*, (PEŠIĆ et al., 2019) implements a real-world situation in a residential space.

QE3. In (BATTISTONI; SEBILLO; VITIELLO, 2019), the authors state that their model is scalable both vertically and horizontally due to the ease of adding layers and nodes within layers. The authors in (PEŠIĆ et al., 2018) cite scalability only in communicating with the MQTT protocol throughout the text and place greater emphasis on the machine learning model. There are no references or citations about scalability in (PEŠIĆ et al., 2019).It is important to point out that none of the corresponding articles experiment (*benchmarks*) in varying the number of devices.

QE4. In (BATTISTONI; SEBILLO; VITIELLO, 2019), the authors propose a *framework* for an architecture in a theoretical way, not providing evaluation or testing metrics. In (PEŠIĆ et al., 2018), the available *benchmark* compares the mean absolute error and the standard deviation in meters of the proposal with other localization algorithms from other *papers*. In (PEŠIĆ et al., 2019), as the authors use the BLEMAT to predict

Table 7 – Comparison of articles and the research proposal

	[1]	**[2]**	**[3]**	**Our Proposal**
Objective	Architecture	Architecture and location location estimation module based om ML	Framework for prediction, detection and data analysis for housing and mobility data based on previous work	Comparison study of traditional placement strategies in IPS based on QoS indicators
Context	Theoretical space-agnostic and application-agnostic	Semi-space-agnostic and context-aware	Semi-space-agnostic	Space-application agnostic
Technologies and location methods	BLE, RSSI	BLE, RSSI	BLE, RSSI	Wi-Fi, RSSI
Testing and simulation	-	Datasets	Datasets and real world application	Datasets and simulation

and detect housing, the authors use their own *dataset* with square root mean error and *Edit Distance on Real Signals*. However, none of the works points out network metrics.

3.5 DISCUSSION

The research scenario is relatively recent, having its first publication in (BATTISTONI; SEBILLO; VITIELLO, 2019) and last publication in 2019 (PEŠIĆ et al., 2019) and a consolidated working group ((PEŠIĆ et al., 2018),(PEŠIĆ et al., 2019)). The technologies used are convergent (RSSI and Bluetooth) and are for generalized contexts. However, researches differs regarding the actual application, having (BATTISTONI; SEBILLO; VITIELLO, 2019) a theoretical approach, and (PEŠIĆ et al., 2018) and (PEŠIĆ et al., 2019), applied. The Table 7 compares our proposal from the presented articles.

Although the objectives are different, the authors demonstrate interest in applications that are somewhat agnostic. However, the articles do not explore or only lightly explore the concept of scalability of the services, nor do they provide a thorough review of the conversion of an OPS to an IPS environment. Also, there is a lack of evaluation metrics, both for their FC applications. However, the authors are more concerned with functionality than system maintainability, leaving other concepts such as scalability out of scope.

In the presented articles, there is no mention of the categories used for the edge mobility model; it is only known that (PEŠIĆ et al., 2018) used real world dataset. Another detail is that in (BATTISTONI; SEBILLO; VITIELLO, 2019), the article's idea is based on theoretical issues, rather than experimentation, which could be better explored through simulation. Unlike the other IPS-FC studies presented, our proposal compares the traditional placement and migration strategies commonly used in OPS to determine whether they can equally suffice for IPS in space agnostic environments.

4 IPS PROPOSAL AND RESEARCH METHODOLOGY

In this section, the proposed methodological paths and the techniques to be used are described.

According to the (MARCONI; LAKATOS, 2012) fundamentals of scientific research, the nature of the work qualifies as applied research, because it aims to generate knowledge for practical applications. As an approach, this study is conceptualized from quantitative analyzes to evaluate the model and to bring data from other studies. The nature of the object of study is considered as descriptive due to the fact that it is a FC study for the context of IPS. As technical procedures, bibliographical research was used, due to the investigation being by articles and experimental because it is a study based on simulations and emulations. Table 8 presents the synthesis.

Table 8 – Search Characterization Synthesis

Nature of Work	Applied Research
Problem Approach	Quantitative Research
Object Nature	Descriptive Research
Technical Procedures	Bibliographic Research and Experimental

4.1 METHODOLOGICAL PROCEDURES

The methodological procedures for this project, as well as the Stages, their objectives, methods used, and results, are presented in Figures 10 and 11. In order to understand the state of the art of IoT, the first Stage used bibliographical research. This Stage was responsible for identifying the overall theory that supports the IoT world.

In Stage two, we refined the research to understand how fog computing is being researched and how positioning systems are implemented in these architectures. Through bibliographical research and reviews, a new objective was met: since the microservice (MS) placements are also being used in fog architectures, identify the traditional MS placement strategies and understand if the mobility of indoor environments is not an issue for these approaches. This Stage enabled the conceptualization of fog computing indoor positioning system models supported by the chosen traditional placements.

In order to demonstrate the conceptual models, it was needed to analyze the open-source simulators that could support our research, as though for phase three. Firstly, it was needed to identify the fog computing simulators, the positioning systems simulators, and then to identify the simulators that could support both fog and positioning systems. For this, we used the previously found articles of the simulators and the iterations of these documents, to detect other authors implementations. As result, we were capable of analyzing the simulator that supported both of the dependencies.

Nonetheless, the proposal was unique in terms of studying placements in indoor environments.

Developing and testing the PS in fog architectures was done in Stage 4. This Stage was needed to identify supporting placement strategies in the simulators as well as the contexts of OPS. These contexts were needed to observe if the simulations were capable of handling mobility parameters ranging from edge device behavior to sensible and critical system parameters and thresholds. Simulations were made to define FCS OPS benchmarks.

For Stage 5, it was noticed that there was a need for further analysis of edge behavior: mobility models. This requirement was necessary due to the fact that the simulators did not support completely indoor environments, missing fog nodes, WiFi range, and context structures besides critical systems, for example, concrete walls, etc. However, even though there was no context-awareness support, the study on the performance evaluation of both environments would be possible. For the findings, it was possible to map the mobility models available and, therefore, develop a thorough problem definition of service migrations. As a main result, the backbone of the research was established: a methodological comparison of indoor and outdoor contexts. During this Stage, the available documentation was used to analyze all of the simulators presented in section 2.7.2. It was discovered that the majority of them lacked theoretical and coding documentation.

Using the methodological approach developed previously in Stage 6, we were capable of implementing the traditional placements in IPS-OPS and benchmarking them as well as the IPS. In order to develop a more realistic set of experiments, we used the workflow presented by (PEŠIĆ et al., 2018) and (PEŠIĆ et al., 2019) through simulation. This careful process originated with the experiments presented in section 5 and the adapted indoor mobility algorithm presented in Appendix A.

Finally, Stage 7 was dedicated to the evaluation and interpretation of the benchmarks proposed. The goal was to provide a detailed comparison of benchmarks, the mobility model, and the incompatibility of placements.Using simulation and the simulation's documentation, it was attested that potentially, there are placement strategies research gaps on indoor positioning systems. These gaps start with the migration time instability of the placement approaches and go up to the overall performance that is not compatible in both environments, as discussed in section 5.

4.2 SIMULATION

Through the simulation, the migration occurs *mobility-induced*, that is, at each user interaction (read as movement), the simulator is responsible for processing the event based on the migration and placement policies configured. As presented in Chapter 2, this study encompasses the study of passive positioning systems using the reference model of the OpenFog Consortium Architecture Working Group (OCAWG, 2017).

4.2.1 Architectures, topologies, and conceptual models

In order to promote deeper analyzes about the scalability of fog architectures, two service placement approaches were selected: edge-ward and clustered. In this present dissertation, the terminology for clustered placement will be used as **horizontal** placement and edgewards, **vertical** placement. It should be noted that migration depends on the placement strategy, and this happens before the simulation starts, so the placement delays are not calculated. Dynamic placements, which are processed during runtime, are not considered since this study targets traditional static placements. The conceptualized model is shown in Figure 12, which uses both placements.

In the clustered placement, the number of clusters (regions) is equal to the proxies, and in each cluster there is the same amount of gateways, except for odd numbers. In this approach, microservices are used. The edgewards approach, however, does not share the cluster groups, instead, the modules are set upward, making, individually each gateway a single process unit and directly dependent on the parent node (proxy).

4.2.2 Vertical placement algorithm

Because devices closer to the network's edge, such as routers and access points, may not be computationally powerful enough to host all application operators, this placement aims to iterate on fog devices towards the cloud while attempting to place remaining operators on alternative devices.The Algorithm 1 represents the edgeward module placement. In this case, a single instance of each MS is deployed to the cloud for each node (GUPTA et al., 2017).

The Algorithm is defined by two steps, iteration and placement, as presented in Algorithm 1. Iterating from the edge device to the cloud, in the physical network topology, the placement of modules is incremented based on the CPU. Module m can only be placed on device d, if all other modules should be placed in the southmost direction (edge direction), in a breadcrumb style.

The placement phase, consists of first checking if an instance of m is already placed on d and then merging the instances and putting them, if possible. Alternatively, a search for parent devices of d that can place the merged instances is triggered. Else, if d or its parent does not have an instance of m, it is then placed on d. In this way, if a parent of d has an instance of m, it was already handled, since it is done leaf-to-root. Having this in mind, multiple instances of a module m can happen at different levels; however, if a device and its parent have instances of a module, the incoming traffic for this module is processed by the device, not by its parent.

In the edgeward placement, there is also fog-to-fog communication, but from tier-1 to tier-2. For this type of placement, the monolithic representation was chosen since, benchmarks for clustered and edgeward approaches are available for outdoor positioning systems. These benchmarks show that vertical placements have inferior performance (GUPTA et al., 2017) when it comes to clustering in OPS.

Algorithm 1: Microservice Edge-ward module placement from (MAHMUD et al., 2022)

```
 1: for $p \in PATHS$ do
 2:   $placedModules \leftarrow \{\}$
 3:   for $fog\_device\ d \in p$ {leaf-to-root traversal} do
 4:     $modulesToPlace \leftarrow \{\}$
 5:     for $module\ w \in app$ {find modules ready for placement on device d} do
 6:       if $P_W \in placedModules$ {if all predecessors of w are placed} then
 7:         $modulesToPlace.push(w)$
 8:       end if
 9:     end for
10:     for $module\ m \in modulesToPlace$ do
11:       if $m \in d.modules$ then
12:         $m' \leftarrow m$
13:         if $CPU_m^{req} \geq CPU_d^{avail}$ {device d does not have CPU capacity to host m}
            then
14:           $m'' \leftarrow merge(m, m')$
15:           $parent \leftarrow d.parent$
16:           while $CPU_m^{req} \geq CPU_d^{avail}$ {find parent device of d for hosting d} do
17:             $parent \leftarrow d.parent)$
18:           end while
19:           $parent.place(m'')$ {device d can host m}
20:           $placedModules.push(m)$
21:         end if
22:       else
23:         $parent.place(m'')$
24:         $placedModules.push(m)$
25:       end if
26:       if $m \in d.parent.modules$ then
27:         if $CPU_m^{req} \leq CPU_d^{avail}$ {if parent has instance of m, will be handled by
            subsequent iterations} then
28:           $d.place(m)$
29:           $placedModules.push(m)$
30:         end if
31:       end if
32:     end for
33:   end for
34: end for
```

4.2.3 Horizontal placement algorithm

As described in the section 2, the horizontal placement aims to distribute the modules among cluster nodes. It uses the same logic as in Algorithm 1, however as it uses the microservice placement, a mapping for the service discovery offered is needed.

4.2.4 Migration event

When a mobility driven event occurs, such as a user changing locations, a migration module is triggered as presented in the Algorithm 3. As the user is always connected to the fog node closest to the top layer (tier 1 - gateway), when moving, each gateway has a load balancer and service discovery entity attached to it, so that if there are multiple instances of the same microservice available in the cluster, it is possible to migrate to the destination node. However, the migration event depends on whether it is moving within a cluster (intracluster) or to another cluster (intercluster). In both cases, two types of communications enable these interactions: edge-to-fog and fog-to-fog communications.

As the Figure 13 shows, if a user moves to another domain (a), the action is intercluster. In this case, there is a need to explore the common accessible point for the old and new gateways (b), involving the upper-tier fog nodes in the migration of modules. Based on the Algorithm 3, in line 8, if the parent of the new and old nearest gateways is the same, then it is an intracluster interaction as described by the Figure 14. In this case, the application modules placed on the old nearest gateway are then placed on the new one through the cluster communication link (fog-to-fog).

4.3 SIMULATION PARAMETERS

In order to provide a more realistic scenario for the experiments, we used the nodes simulation parameters in iFogSim v2 for the Crowd Sensing and Cardiovascular Health Monitoring experiments (MAHMUD et al., 2022). The fractional selectivity in the simulator means the packet loss, which in this study is omitted, is set to 1.0.

4.4 APPLICATION MODEL

Figure 15 describes the directed acyclic graph (DAG) of the application model based on (PEŠIĆ et al., 2018) and (PEŠIĆ et al., 2019) with the module relationships shown on the edges. The blue circles represent the interactions of the edge devices (input and output), and the red circles represent the microservices placed in the fog and cloud. Because it is a simulation, the communication protocols are omitted. Application models can also be thought of as application workflows, because each module follows

Algorithm 2: Microservice Clustered module placement from (MAHMUD et al., 2022)

```
1: for $cluster \in clusters$ do
2:   for $p \in PATHS$ do
3:     $placedModules \leftarrow \{\}$
4:     for $fog\_device\ d \in p$ {leaf-to-root traversal} do
5:       $modulesToPlace \leftarrow \{\}$
6:       for $module\ w \in app$ {find modules ready for placement on device d} do
7:         if $P_w \in placedModules$ {if all predecessors of w are placed} then
8:           $modulesToPlace.push(w)$
9:         end if
10:      end for
11:      for $module\ m \in modulesToPlace$ do
12:        if $m \in d.modules$ then
13:          $m' \leftarrow m$
14:          if $CPU_m^{req} \geq CPU_d^{avail}$ {device d does not have CPU capacity to host m}
             then
15:            $m'' \leftarrow merge(m, m')$
16:            $parent \leftarrow d.parent$
17:            while $CPU_m^{req} \geq CPU_d^{avail}$ {find parent device of d for hosting d} do
18:              $parent \leftarrow d.parent)$
19:            end while
20:            $parent.place(m'')$ {device d can host m}
21:            $placedModules.push(m)$
22:            $registerNode(parent)$
23:          end if
24:        else
25:          $parent.place(m'')$
26:          $placedModules.push(m)$
27:          $registerNode(parent)$
28:        end if
29:        if $m \in d.parent.modules$ then
30:          if $CPU_m^{req} \leq CPU_d^{avail}$ {if parent has instance of m, will be handled by
             subsequent iterations} then
31:            $d.place(m)$
32:            $placedModules.push(m)$
33:            $registerNode(d)$
34:          end if
35:        end if
36:      end for
37:    end for
38:  end for
39: end for
```

Algorithm 3: Mobility Management in iFogSim v2 adapted from (MAHMUD et al., 2022)

```
1: $\delta \leftarrow maxDistance$
2: for $NG \in m.parents()$ do
3:     if $haversine(L_{NG}, L_m) \leq \delta$ then
4:         $\delta \leftarrow haversine(L_{NG}, L_m)$
5:         $p \leftarrow NG$
6:     end if
7: end for
8: if $p' \neq p$ {when the parent differs} then
9:     $pMM \leftarrow p'.placedModulesOfMobileDevice(m)$
10:    if $p.cluster == p'.cluster$ then
11:        $pushModulesFromTo(p', p, pMM)$
12:    else
13:        $common \leftarrow null$
14:        $n \leftarrow p.pathsToCloud$
15:        $n' \leftarrow p'.pathsToCloud$
16:        for $node \in n$ do
17:            for $node' \in n$ do
18:                if $node = node'$ then
19:                    $common \leftarrow node$
20:                    $break$
21:                end if
22:            end for
23:        end for
24:        $pushModulesFromTo(p'.modules, node.modules, pMM)$
25:        $pushModulesFromTo(node.modules, p.modules, pMM)$
26:    end if
27:    $m.parent \leftarrow p$
28:    $p.modules \leftarrow pMM$
29:    $p'.terminateModules(m, pMM)$
30: end if
```

or is followed sequentially by the others. For each module, simulation values were assigned, such as RAM and CPU usage, input and output size, as shown in Table 11.

4.5 DATA

The location of the nodes' geographic data, as well as the user movement datasets, were based on the EUA dataset [1]. Interactions are paths that a user takes within an area. Each interaction counts as a geographic location containing latitude and longitude, and for each user, 20 interactions are generated, and steps can be read, as represented in Figures 20 and 17 by the blue markers.

[1] https://github.com/swinedge/eua-dataset

Table 9 – Nodes simulation parameters

	Cloud	Proxy	Gateway	Smartphone
Speed (MIPS)	4480	2800	2800	200
RAM (GB)	16	4	4	2
Uplink (MBPS)	100	10	50	100
Downlink (MBPS)	100	20	100	200

Table 10 – Latency simulation parameters

From	To	ms
Proxy	Cloud	100
Gateway	Proxy	4
Proxy	Proxy	2*
Device	Gateway	2

*: cluster link latency

Table 11 – Simulation parameters for application modules

	RAM *	Input **	Output **	CPU Length ***
Client	0.1	0.5	0.5	1000
Kalman F.	0.5	0.5	0.5	2000
Distance Calc.	0.5	0.5	0.5	1800
Trilat.	0.5	0.5	0.5	1000
Position Est.	0.5	0.5	0.5	500
Data An.	2	0.5	0.5	500

*:GB, **: MB, ***: MIPS

The user dataset, which contains such interactions, has predefined values (manual) and is generated by the modified algorithm presented in the APPENDIX A which simulates the possible interactions that users may have in the room within their limits, depending only on your scenario.

4.6 FOG CONFIGURATIONS

The nodes are defined as belonging to the same region by the identifier code of a node in the upper layer; thus, the parent node of all gateways is a proxy, and that of a proxy is the cloud, in accordance with the architecture's hierarchy.The proxies are kept in the central region of the region in indoor experiments and in the first row in the OPS-IPS comparison study.

As for the placements, in the clustered one, the number of clusters is the same as the number of proxies, which is also considered a region, so that gateways from the same cluster can communicate. On the other hand, in the vertical placement, the clusters are considered only a matter of order. In this placement, the gateways in the same cluster do not communicate, but only through proxies (immediate parent node).

4.7 CONTEXTS

We conducted various sets of experiments in order to provide a more thorough analysis of the placements' performance in the various environments.These simulations include IPS using clustered and edgeward placements with iterations of different numbers and configurations of nodes, one base case for clustered OPS placement, and also a comparison of each environment based on the clustered placement. The adaptation process from real to virtual (digital) environments aimed to preserve scale and mobile patterns.

4.7.1 Indoor Positioning Systems scenarios

To provide a more in-depth study of architectures within *indoor* contexts, two different experiments were generated.

The application case studies are spatial application scenarios based on environments at the Federal University of Santa Catarina: Department of Sanitary Engineering (ENS) and the Department of Informatics and Statistics (INE), both from the Technological Center (CTC), described in the next two subsections. In both scenarios, smartphones (users) can move in the areas described by the scenarios. To simulate their movements, data was dynamically generated based on a predefined dataset (MAHMUD et al., 2022). It is important to note that the two scenarios were selected due to their spatial divergence, i.e., format and entry points.

Since the iFogsim by default configured for OPS, code changes were necessary for generating user movement in indoor environments, setting starting points, and locating edges and fog devices. The formation of the movement of users was based on the mobility principles *indoor*, following the principles of the Gauss-Markov mobility model. This adapted random mobility algorithm, presented in Appendix A, for indoor environments was adapted from the one available for outdoor environments in the simulator. It was necessary to change the entry points accordingly to the scenario created, as well as the limit region of the area. This definition can be read as boundary limits. These limits were adjusted accordingly to the scenarios. Also, the step distance was reduced so that each user "walked" a few meters instead of many meters. We preserved an average of meters for the IPS scenarios.

4.7.1.1 Department of Sanitary Engineering (ENS)

The proposed scenario is the area of the Sanitary Engineering environment located in the Technological Center, at the Federal University of Santa Catarina (UFSC) whose area is 310m^2. There are two entry points: one to the north and one to the south. For users, there are four north and one south, while there are eight north and seventeen south. Figure 17 presents the movement studied with 5 and 25 users. And for better visualization, a heatmap is presented in Figure 18.

4.7.1.2 Department of Informatics and Statistics (INE)

In this scenario, the perimeter of 104m is discussed, whose equivalent area is 460m^2. This scenario is located in the area of Informatics and Statistics (INE) in the Technological Center, at the Federal University of Santa Catarina (UFSC). As for the user/meter ratio, for scenario 5, 20.8m^2 per person, and for scenario 25, 4.16m^2. The reference dataset for 5 and 25 users used only one reference. Figure 20 presents the movement studied with 5 and 25 users. And for better visualization, a heatmap is presented in Figure 21.

4.7.1.3 Experiment one - E1

This experiment was considered in order to verify how much the architectures differ depending on the number of nodes per region, relating to the proportion rate between gateways and proxies in order to verify the possibility of patterns in blocks.The first experiment consists of exploring the number of regions with different numbers of gateways (Table 12).

Table 12 – Correlation of fog nodes in experiment one

Gateways	Proxies	**Total**
4	1	5
	2	6
8	2	10
	3	11
12	2	14
	3	15
16	3	19
	4	20

For a better study based on the system's performance, the nodes were placed based on the average distance ratio among them. Figures 25, 24, 23 and 22 present the configurations in the demonstrated experiment for the ENS scenario. The configuration of Figure 22 and 26 were also used in E2, but further details are provided in the related section below.

Based on 22, in order to compare the architectures scalability varying on the number of fog nodes, the other configurations (Figure 23 - 25), were reduced. Firstly, it was configured with 20 nodes (16 gateways - 4 regions) and each node had a distance of 4 meters intracluster; secondly, 6 meters (16 gateways - 3 regions); thirdly, 8 meters (12 gateways - 3 regions); and so forth and so on. For the E1, the rule for picking the proxy was the fog node nearest the bounds. This selection was made to respect the OPS EUA dataset proxy picking. The same process was made in scenarios INE and ENS.

Figures 29, 28, 27 and 26 present the configurations in the demonstrated scenario. As such, each color represents a cluster, and the fog node tier description is also described nominally. In the presented scenarios, the proxies are preserved in the center of their cluster region. As the setting of fog nodes was presented above, for the INE fog nodes average distance ratio is as follows: Figure 26 was 4m and 5m, 27 was 6m and 7m, 28 was 8m and 9m and Figure 29 was 9m and 9m.

4.7.1.4 Experiment two - E2

Because of the wide range of data in E1, it was decided to create another experiment. In this, the same number of total nodes was maintained for each proposed architecture (Table 13). The objective proposed here is to verify the uniform variation of each configuration. In this way, the focus is on a specific comparison of approaches by region.

Table 13 – Correlation of fog nodes in experiment two

Gateways	Proxies	**Total**	Nodes by Region
19	1	20	20
18	2	20	10
17	3	20	≈ 7
16	4	20	5

Using the previously presented configuration of 22, Figures 30 and 31 show the configurations in the experiment presented. Differently from the E1, the rule for proxy setting was to divide equally the proportion of fog nodes on each cluster.

Figure 32 show the configuration of a monocluster in the scenario presented, and Figure 33, a double cluster. The selection of proxies was different from the ENS environment. In this case, the proxies were placed respecting the centrality of the clusters, since this scenario has a fixed number of fog nodes.

4.7.2 Outdoor Positioning Systems scenarios

Outdoor positioning systems can be considered the traditional ones. To enable further analysis, we provided an experiment using clusterization to understand how this orchestration works when it comes to scalability behavior. For this experiment, we based our references on the case study of Crowd-sensed Data Collection presented in (MAHMUD et al., 2022). The application model was simplified, as shown in Figure 34.

User mobility in outdoor environments, such as the one presented, can depend on a number of variables. To ensure most of these possibilities, this section exhibits the environments created. As a proposal, we investigate the possibility of a clustered-fog configuration with a faster application response time than the default EUA dataset, taking into account the following mobility parameters: speed, direction, and grouping.Besides the presented parameters, the architecture should also perform adequately for the variation in the number of users.

The presented architectures are divided into three layers: edge, fog, and cloud.We present a modern scenario in which the edge consists of users sending their geolocation, through an application on their smartphones, which are connected to the fog layer. The fog layer contains clustered nodes located across the area. In this layer, there are computing nodes that execute tasks and orchestrators that delegate the application flow and are also connected to the cloud. And finally, the cloud is represented as a single datacenter.

4.7.2.1 Fog nodes configurations

The cluster configurations are the variations of the quantity of clusters which were eight, twelve (default EUA dataset) and twenty eight cluster groups, and also on the number of fog nodes for each cluster. All of the clusters were generated based on the default, preserving the centrality of the proxy (orchestrator node) in the regions, manually. Figure 35 depicts the configurations on a map.

4.7.2.2 Users mobility

The cluster configurations differ in the number of clusters, which were eight, twelve (default EUA dataset), and twenty-eight, as well as the number of fog nodes for each cluster.All of the clusters were manually generated based on the default, preserving the centrality of the proxy (orchestrator node) in the regions. Figure 35 depicts the configurations on a map.

To evaluate the cluster configurations service performance, two different sets of users were tested: eight and fifty users. The eight-user set was configured as presented in Table 14. The fifty-users set is a repetition of the eight-users set, which means that the last two users (49th and 50th) were the first and second from the copied dataset, respectively.

Table 14 – Relation of users in eight-users dataset

Repetitions	Speed	Start
1	Fast	NW
2	Medium	S
2	Medium	N
3	Slow	SE

Table 15 – Simulation parameters for the devices

	Cloud	Proxy	Gateway	Smartphone
Speed (MIPS)	4480	2800	2800	200
RAM (GB)	16	4	4	2
Uplink (MBPS)	100	10	50	100
Downlink (MBPS)	100	20	100	200

4.7.3 Comparison of indoor to outdoor environment placement strategies

Indoor positioning systems consist of locating devices on a smaller scale compared to outdoor environments. Not only do these systems differ in their geographic spaces, but also in the number of objects to be located. As a result, in order to develop experiments based on mobility for indoor systems, a standardized mobility experiment, as presented in this subsection, was required.For this environment, we used the clustered placement strategy, since it was a better approach for the OPS environments, as described in section 4.7.2. The application model and simulation parameters used are described in section 4.

4.7.3.1 Fog nodes configurations

We created two different configurations for each environment in order to evaluate the performance of the configurations.Every fog node has the same computing power, as presented in Table 15. In our experiments, we developed two environments: indoor and outdoor. For the outdoor environment, we chose to use 16 clusters with 128 fog nodes, with an average of 128/16 = 8 nodes per cluster, and 8 clusters with 64 fog nodes, with an average of 64/8 = 8 as presented in Figure 37. The nodes locations were also changed to follow a smart context logic, such as smart city and smart home, that could be reduced and compared to an IPS, in that they are both sequential.

For the indoor environments, there were 4 nodes in 2 clusters with 4/2 = 2 nodes per cluster and 1 cluster with 2 fog nodes, as presented in Figure 38. It is important to note that there is a lack of IPS area definition, since these systems can be considered anywhere from a smart home to a large smart parking lot or even airports.

Like the default dataset, we had two types of fog nodes: gateways and proxies.

The proxies are defined as the first nodes near the entry point for the sake of simplicity. For both environments, the timing and spacing of interactions were kept because, even though the two environments cover different distances in meters or kilometers, the interactions of the users are the same in meters because the devices in both environments are carried by people.

Also, when it comes to investigating positioning systems, the radius of the fog nodes (coverage) is another factor that is not usually cited. With this in mind, we define the average radius coverage (ARC). This variable is defined by the average distance of the intracluster and intercluster divided by two. For the OPS environment, as each fog node was placed with an average distance of 100m intracluster and 60–70 m intercluster, the average radius coverage was approximately $40m$. As for the IPS, in configuration A, the intracluster distance was 80m and the intercluster distance was 200m, giving an ARC of $140m$. In configuration B, as there is only one cluster, the average distance of each fog node was 200m, resulting in an ARC of $100m$.

To investigate the performance equivalence of the clustered micro-service horizontal placement strategy in OPS and IPS, we used two sets of fog nodes for the IPS (4 nodes in two clusters and 2 nodes in one cluster), one set of fog nodes for OPS (130 nodes in 16 clusters), and three users (5 and 25) for each mobility pattern, as explained in the subsections below.

The different numbers of users were used over the interactions to analyze the scalability performance of each environment during the bootstrapping effect, i.e., at the beginning of each experimentation. The following aspects involved in both environments were equal or equivalent:

- Distance for each interaction: as the core of the research is the IPS, we prioritize the similarity of interactions on each mobility pattern.
- Fog node computing resource: is equal to both the IPS and OPS scenarios.
- Geographical space: both environments are rectangular with a single entry point.

4.7.3.2 Scenarios

To evaluate the equivalence of the performance for both environments (indoor and outdoor), we define three mobility patterns, or in this case, called scenarios. Each of them is responsible for overloading a specific area to investigate the migration of services, overall performance, and the existence or nonexistence of correlation based on the context.

For each mobility pattern in the environments, the number of interactions as well as the meters per interaction are different. This happens because the number of interactions was calculated using the meters per interaction from the OPS. E.g.: in scenario one (cross area), if the meters per interaction are $5m$, then each interaction in the IPS environment should also be $5m$, and since it has a lesser total area than the OPS, the number of interactions is 34 and so the users roamed $170m$.

The first of the scenarios is cross-area (CA) which draws the area diagonally in the stated rectangle region. It aims to overload every cluster in a single line. The meters per interaction is $5m$. The Table 16 defines the relation of interactions and meters by environment. Based on Table 16, the Figure 39 presents the exchangeability of the environments.

Table 16 – Relation of interactions, meter and environments of scenario one

Interactions	Meters	Environment
34	170	Indoor
310	1550	Outdoor

The second, line through (LT) brings the computing need to the first portion of both environments. The objective is to overload the first cluster (s) evenly. Here, the meters per interaction are $8.2m$. Table 17 presents the equivalence used for interactions, in meters per environment. Based on Table 17, the Figure 40 presents the exchangeability of the environments.

Table 17 – Relation of interactions, meter and environments of scenario two

Interactions	Meters	Environment
25	205	Indoor
220	1800	Outdoor

Inline (IL) is the third mobility pattern, and it brings the computing need to the first portion of both environments. The objective is to overload the first cluster (or clusters) evenly. Here, the meters per interaction are $8.2m$. Table 18 presents the equivalence used for interactions, in meters per environment. Based on Table 18, the Figure 41 presents the exchangeability of the environments.

4.8 DISCUSSION

One of the main challenges was finding the theoretical background for the traditional placements in the simulators. iFogSim describes both edgeward and clustered placements using a pseudocode in the research paper. The other simulator experimented was YAFS. This simulator is currently in version 0.5, and it is noticed there is a lack of documentation as well as theoretical background for the default service placements. Also, Python version problems happened during the experimentation with the environments. The iFogSim v1, however lacked mobility features, so it was only possible to simulate static environments. Its successor, iFogSim v2 introduced the possibility of evaluating the performance of positioning systems.

Table 18 – Relation of interactions, meter and environments of scenario three

Interactions	Meters	Environment
17	107	Indoor
143	900	Outdoor

When it comes to mobility models, the random models (RM) are used in the majority of articles, to evaluate the performance of the locationing system. Additionally, using this model, it is possible to research hypothetical mobility events. Still about edge behavior, signal strength and location precision are important aspects of every PS. RSSI, for example, uses the measurement of the power signal present in a received radio signal. In this strategy, the stronger the signal, the better the precision. However, there are many factors that affect the precision besides the user's distance, such as IPS, walls, location, and even the number of users. These noise variables were minimized in the experiment.

Currently, the encountered simulators have received new updates for the first semester of 2023. However, since each simulator have their own code implementation, it would be interesting to enable comparative studies with different fog-based positioning system simulators. Yet in the OSC, new projects like Kube Edge (XIONG et al., 2018). Also A community hub could help developers find projects that make possible the development of Fog IPS and newer and more specialized placement strategies. Also A community hub could help developers find projects that make possible the development of Fog IPS and newer and more specialized placement strategies.

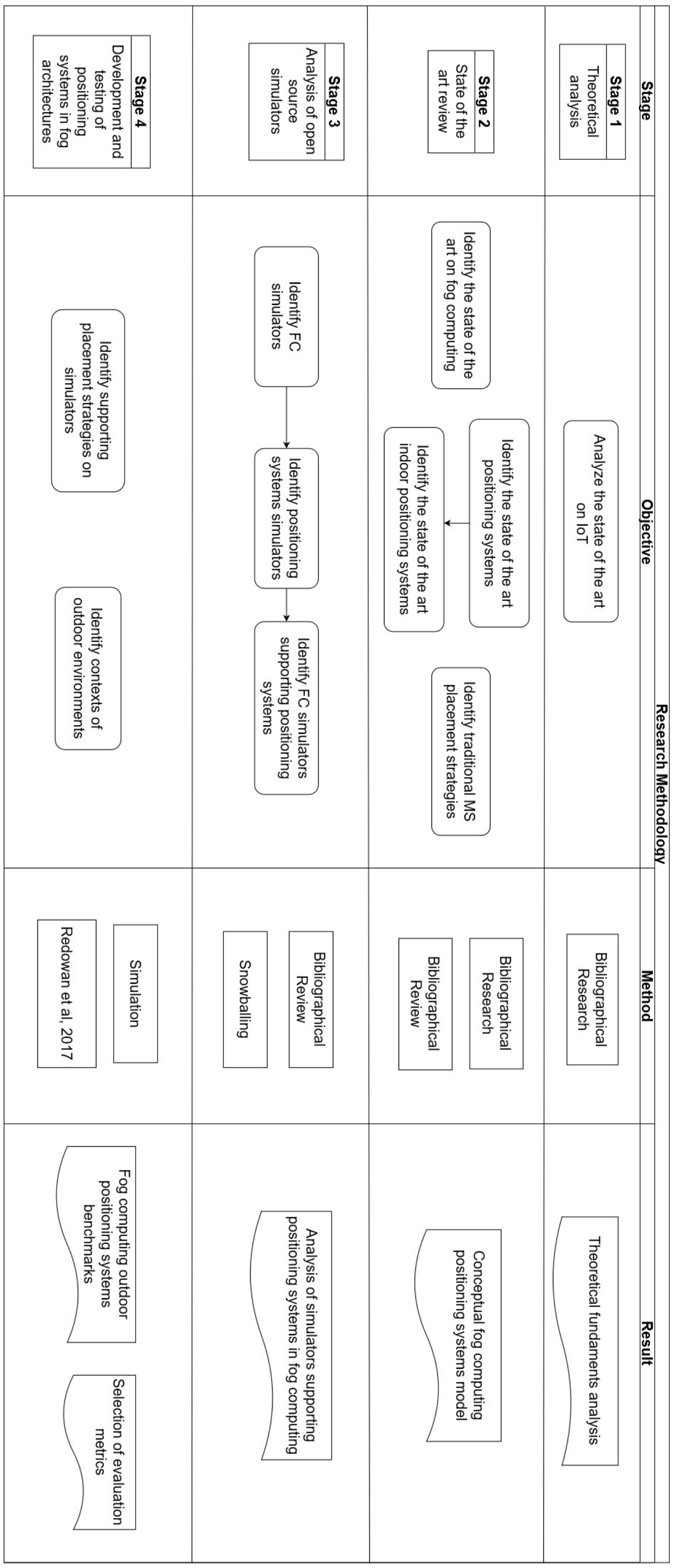

Research Methodology
Stage
Objective
Method
Result
Stage 1
Theoretical analysis
Analyze the state of the art on IoT
Bibliographical Research
Theoretical fundaments analysis
Stage 2
State of the art review
Identify the state of the art on fog computing
Identify the state of the art positioning systems
Identify the state of the art indoor positioning systems
Identify traditional MS placement strategies
Bibliographical Research
Bibliographical Review
Conceptual fog computing positioning systems model
Stage 3
Analysis of open source simulators
Identify FC simulators
Identify positioning systems simulators
Identify FC simulators supporting positioning systems
Bibliographical Review
Snowballing
Analysis of simulators supporting positioning systems in fog computing
Stage 4
Development and testing of positioning systems in fog architectures
Identify supporting placement strategies on simulators
Identify contexts of outdoor environments
Simulation
Redowan et al, 2017
Fog computing outdoor positioning systems benchmarks
Selection of evaluation metrics

Figure 10 – Research methodology procedures (Stages 1 - 4)

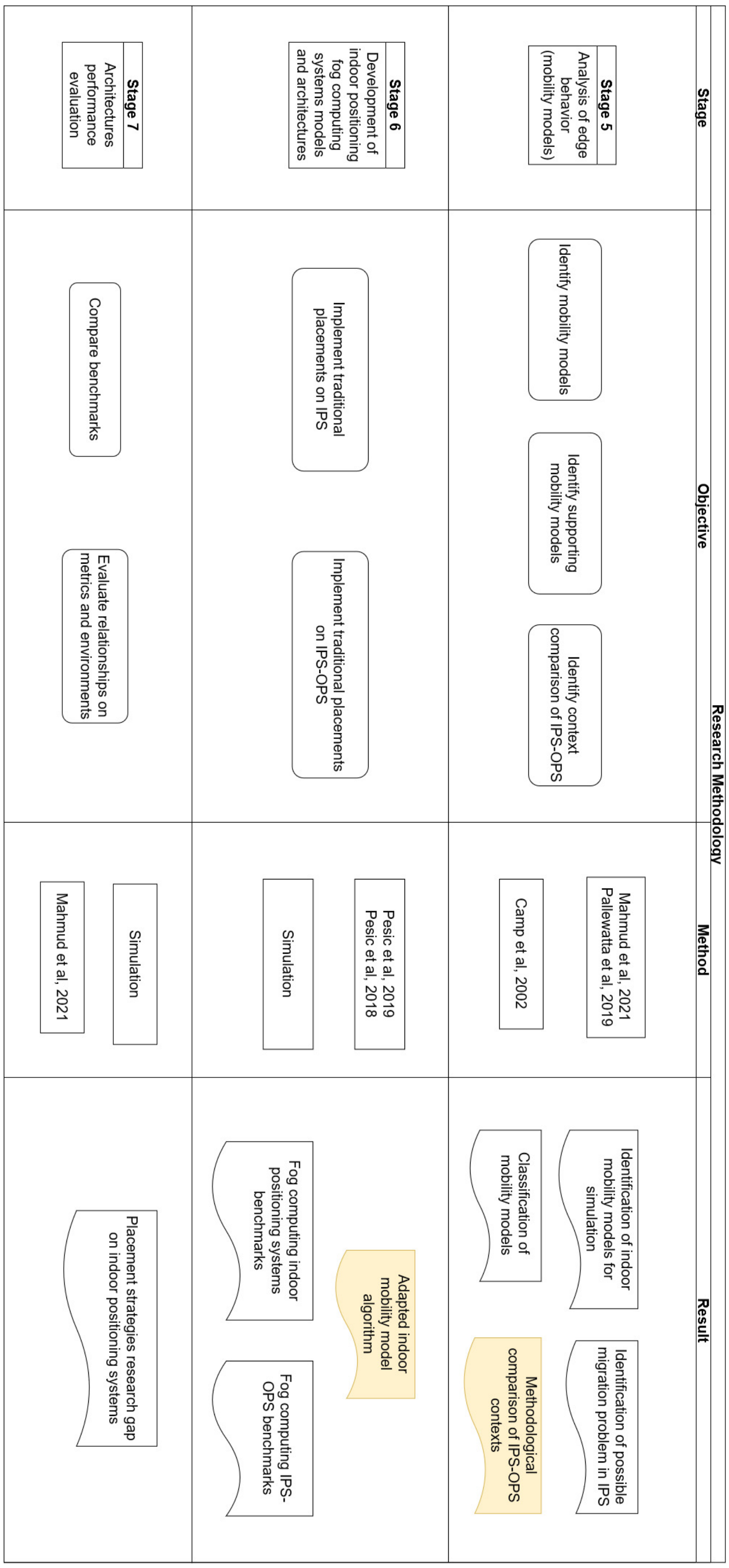

Research Methodology
Stage
Objective
Method
Result
Stage 5
Analysis of edge behavior (mobility models)
Identify mobility models
Identify supporting mobility models
Identify context comparison of IPS-OPS
Mahmud et al, 2021
Pallewatta et al, 2019
Camp et al, 2002
Identification of indoor mobility models for simulation
Classification of mobility models
Identification of possible migration problem in IPS
Methodological comparison of IPS-OPS contexts
Stage 6
Development of indoor positioning fog computing systems models and architectures
Implement traditional placements on IPS
Implement traditional placements on IPS-OPS
Pesic et al, 2019
Pesic et al, 2018
Simulation
Adapted indoor mobility model algorithm
Fog computing indoor positioning systems benchmarks
Fog computing IPS-OPS benchmarks
Stage 7
Architectures performance evaluation
Compare benchmarks
Evaluate relationships on metrics and environments
Simulation
Mahmud et al, 2021
Placement strategies research gap on indoor positioning systems

Figure 11 – Research methodology procedures (Stages 5 - 7)

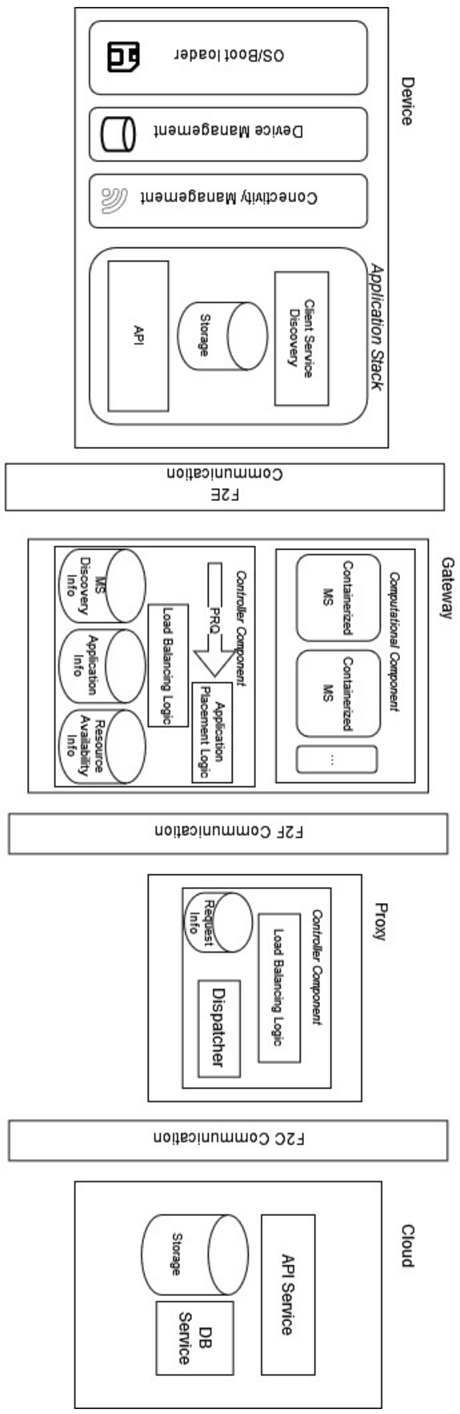


Figure 12 – Conceptual model of architectures

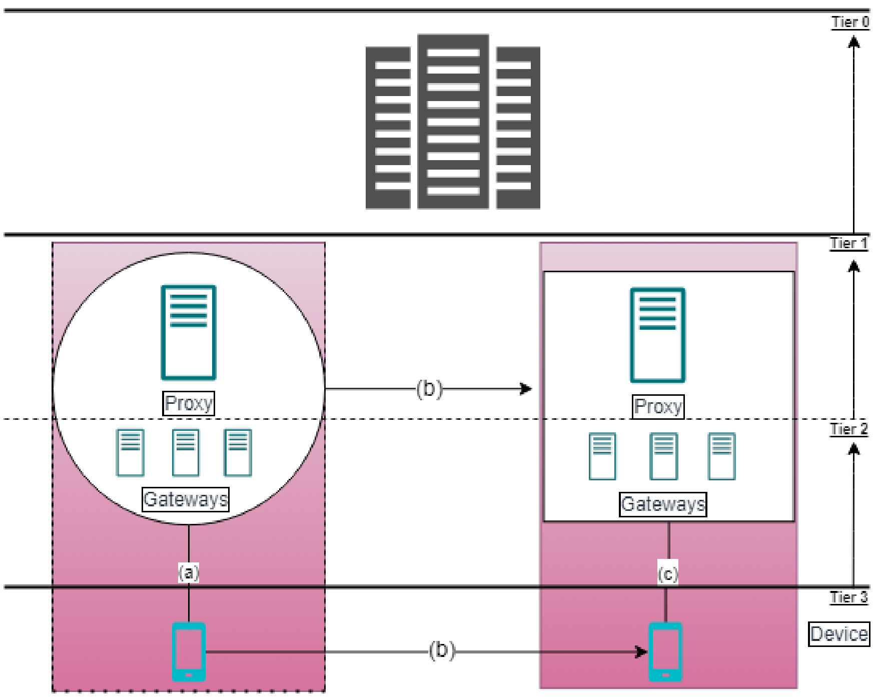


Figure 13 – Migration intercluster in clustered approach

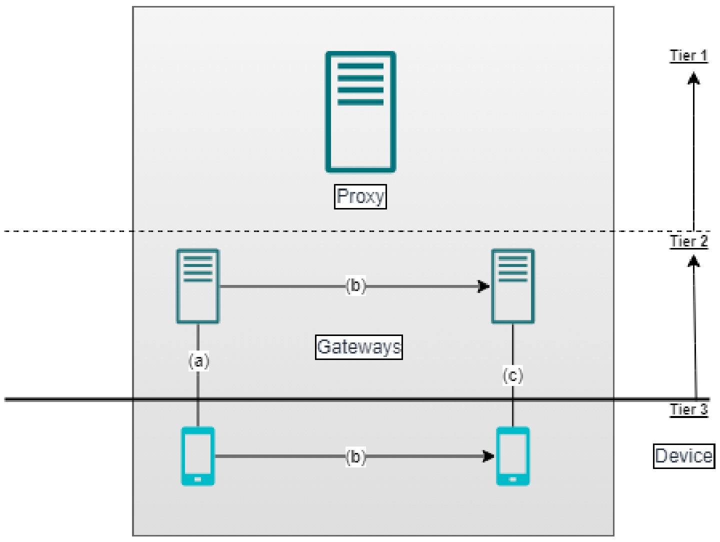


Figure 14 – Migration intracluster in clustered approach

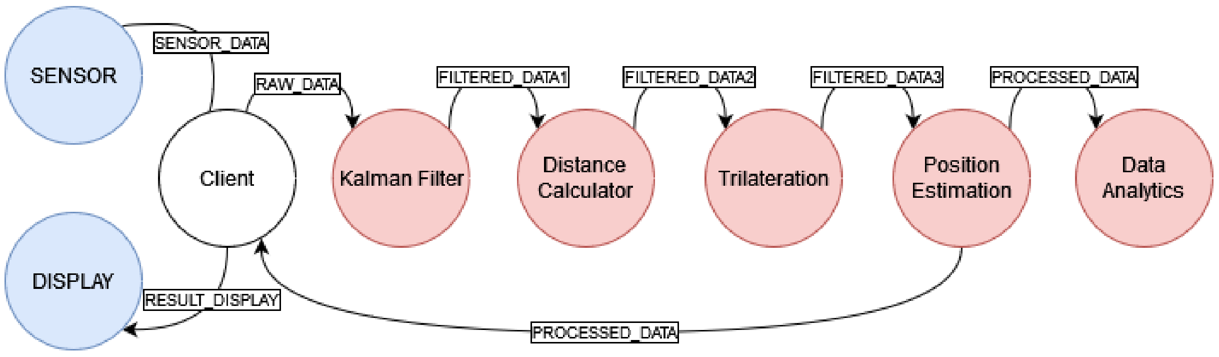


Figure 15 – Application model for the indoor case studies based on the modules present in (PEŠIĆ et al., 2018) and (PEŠIĆ et al., 2019)

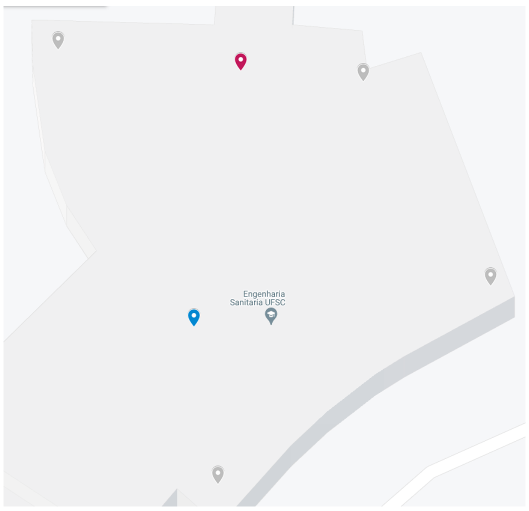


Figure 16 – Location of the entries and boarders of the proposed ENS scenario

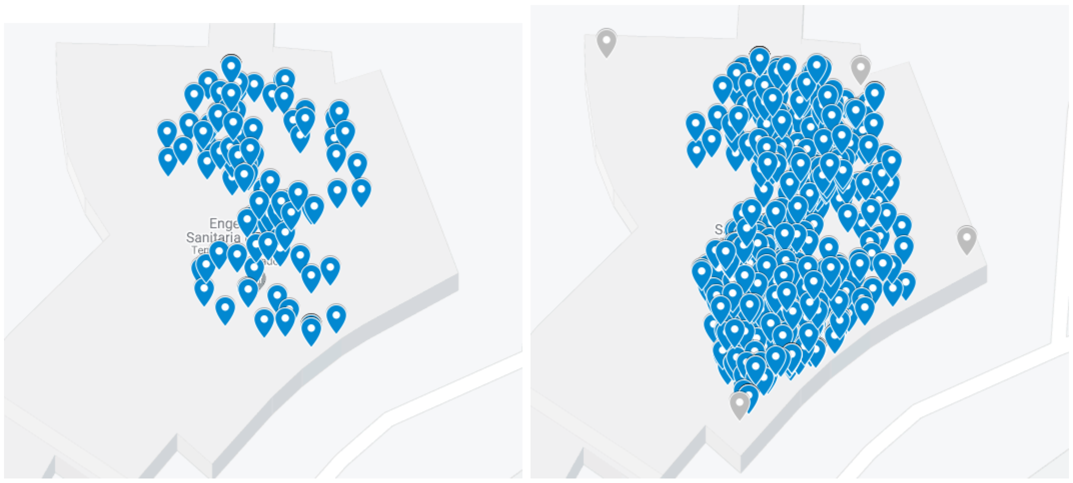


Figure 17 – Moving 5 users and 25 in the ENS scenario

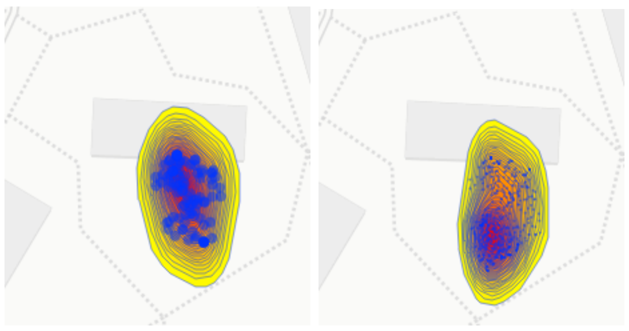


Figure 18 – Heatmaps for 5 users and 25 in the ENS scenario

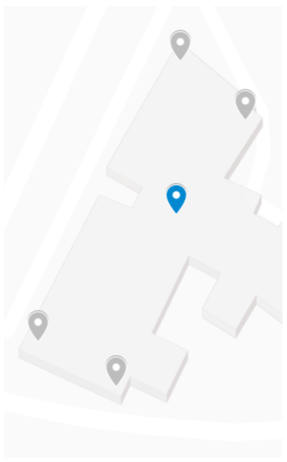


Figure 19 – Location of the entries and boarders of the proposed INE scenario

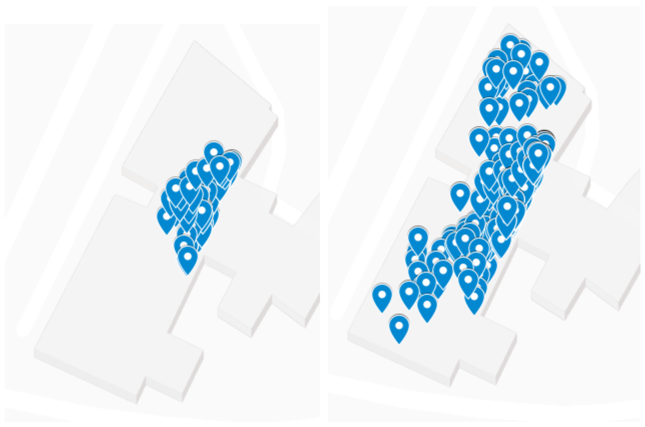


Figure 20 – Moving 5 users and 25 in the INE scenario

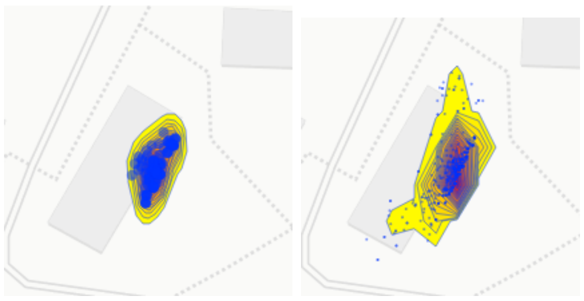


Figure 21 – Heatmaps for 5 users and 25 in the INE scenario

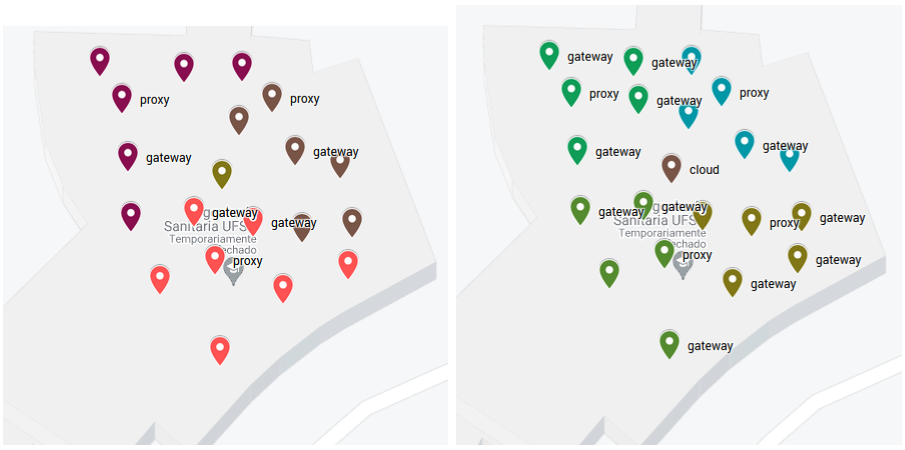


Figure 22 – 16 gateways with 3 proxies on the left and 4 on the right in the ENS scenario

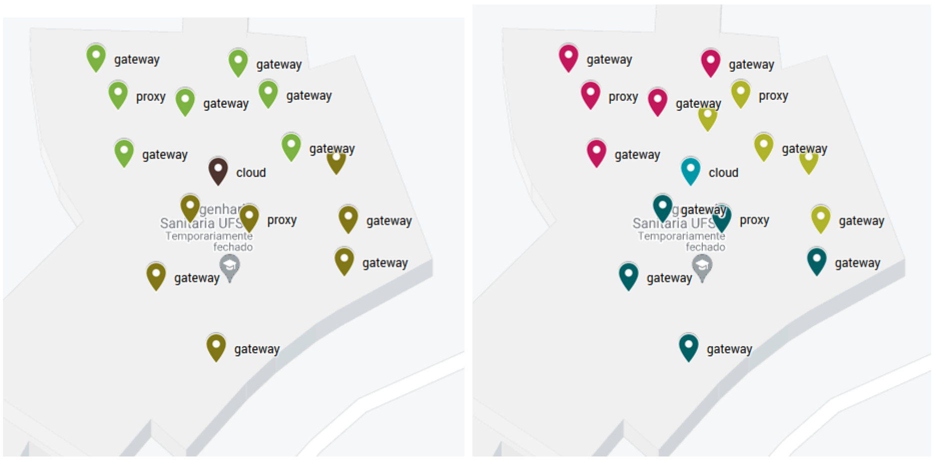


Figure 23 – 12 gateways with 2 left and 3 right proxies in the ENS scenario

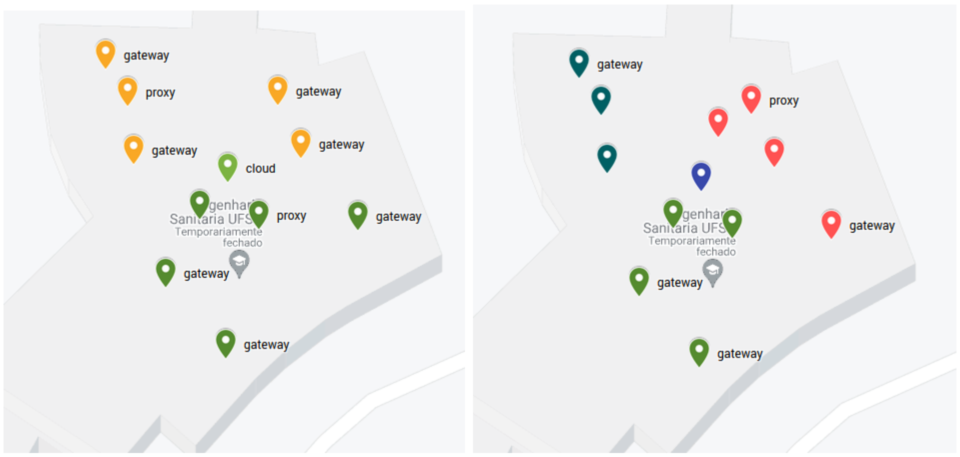


Figure 24 – 8 gateways with 2 left and 3 right proxies in the ENS scenario

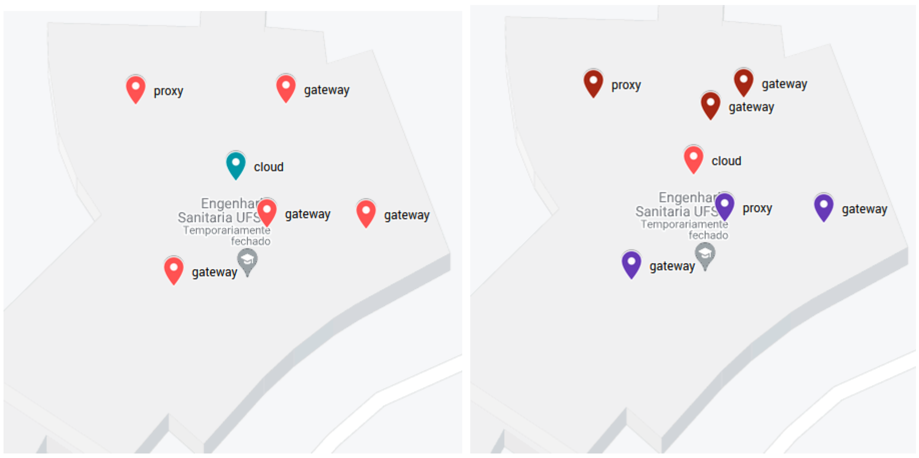


Figure 25 – 4 gateways with 1 proxy on the left and 2 on the right in the ENS scenario

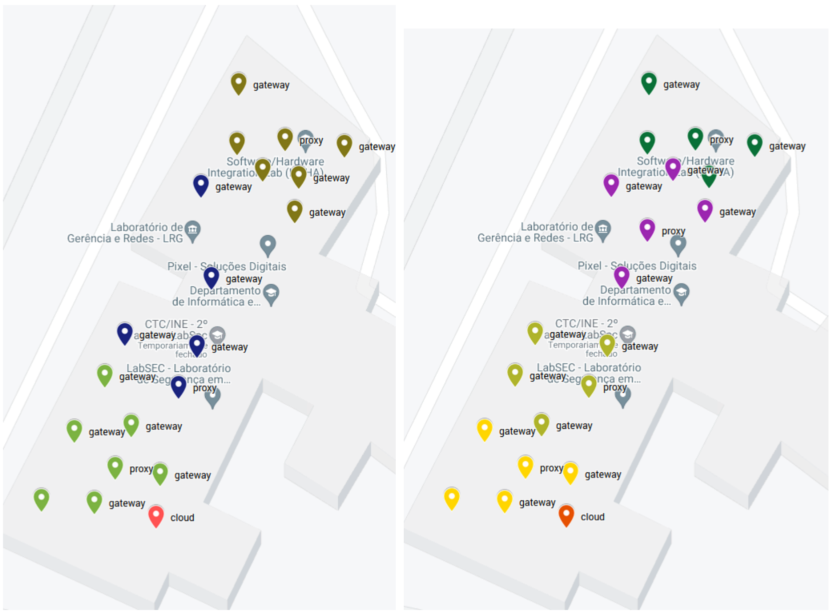

gateway
proxy
Software/Hardware Integration Lab (LISHA)
Laboratório de Gerência e Redes - LRG
Pixel - Soluções Digitais
Departamento de Informática e...
CTC/INE - 2º andar - LabSEC
LabSEC - Laboratório de Segurança em...
cloud

Figure 26 – 16 gateways with 3 proxies on the left and 4 on the right in the INE scenario

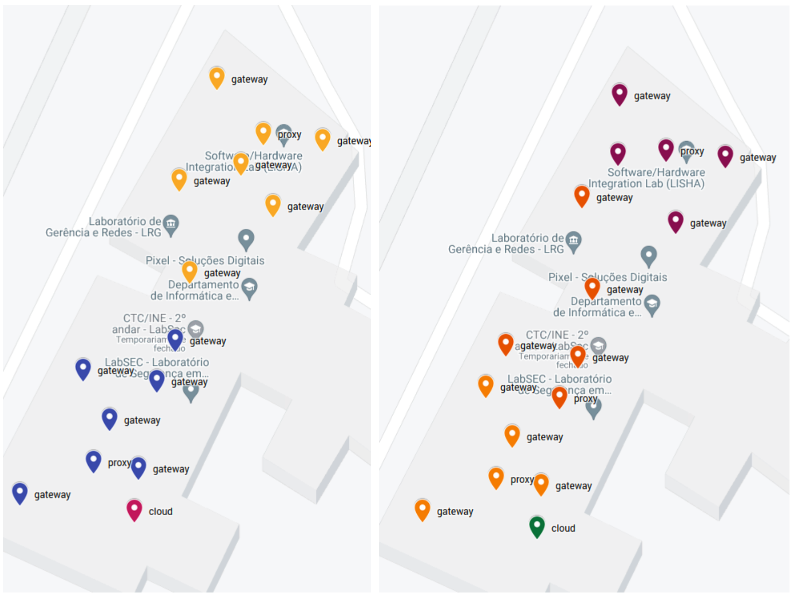

gateway
proxy
Software/Hardware Integration Lab (LISHA)
Laboratório de Gerência e Redes - LRG
Pixel - Soluções Digitais
Departamento de Informática e...
CTC/INE - 2º andar - LabSEC
LabSEC - Laboratório de Segurança em...
cloud

Figure 27 – 12 gateways with 2 left and 3 right proxies in the INE scenario

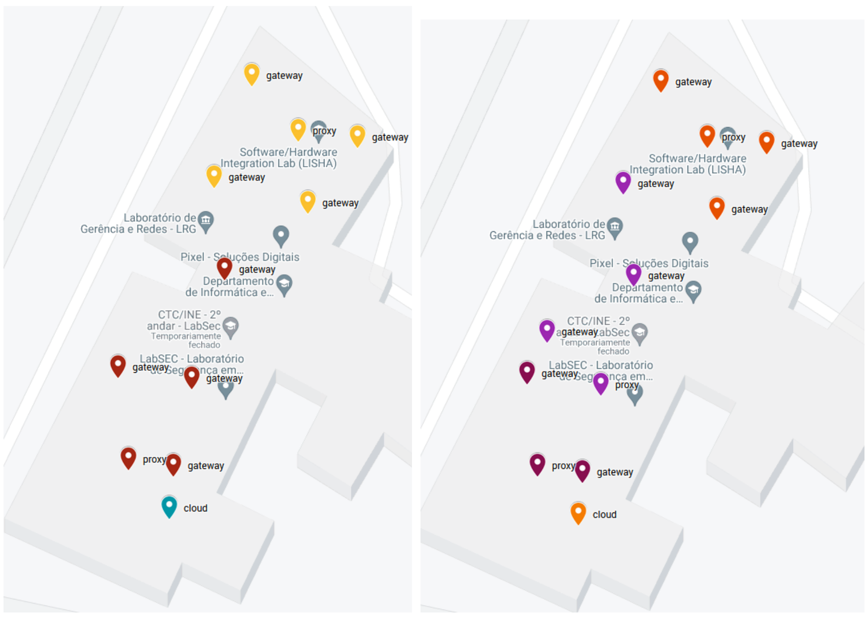

gateway
proxy
gateway
Software/Hardware Integration Lab (LISHA)
gateway
gateway
Laboratório de Gerência e Redes - LRG
Pixel - Soluções Digitais
gateway
Departamento de Informática e...
CTC/INE - 2º andar - LabSec
Temporariamente fechado
LabSEC - Laboratório de Segurança em...
gateway
gateway
proxy
proxy
gateway
cloud

Figure 28 – 8 gateways with 2 left and 3 right proxies in the INE scenario

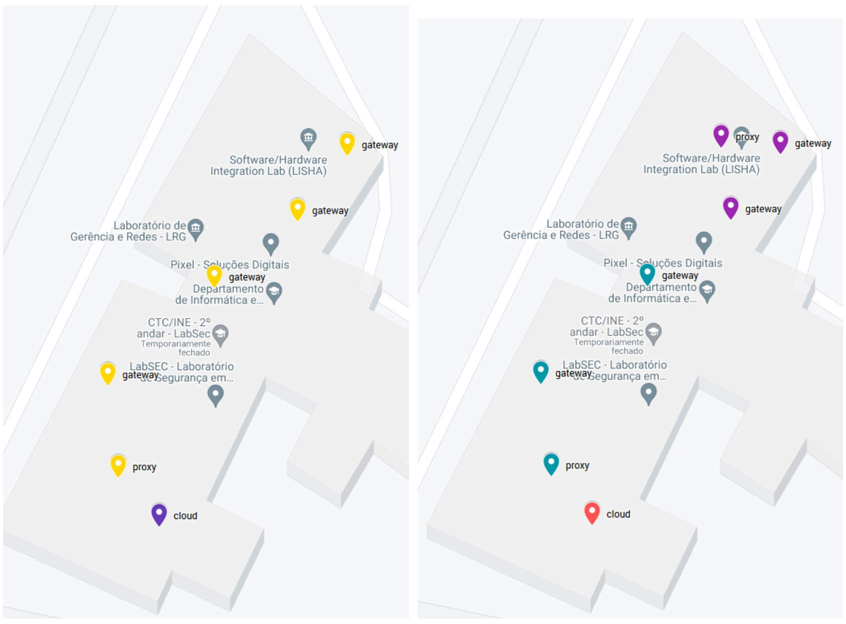

proxy
gateway
Software/Hardware Integration Lab (LISHA)
gateway
Laboratório de Gerência e Redes - LRG
Pixel - Soluções Digitais
gateway
Departamento de Informática e...
CTC/INE - 2º andar - LabSec
Temporariamente fechado
LabSEC - Laboratório de Segurança em...
gateway
proxy
cloud

Figure 29 – 4 gateways with 1 proxy on the left and 2 on the right in the INE scenario

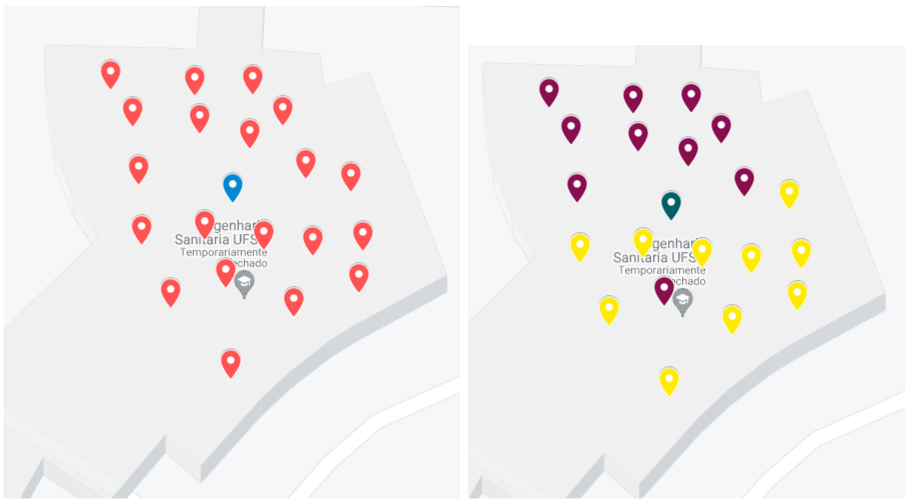


Figure 30 – 20 nodes with 1 region on the left and 2 on the right in the ENS scenario on E2

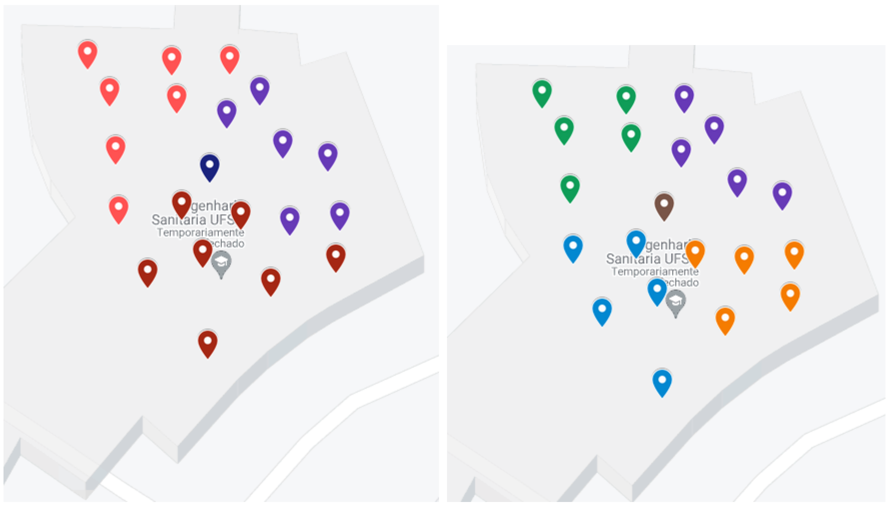


Figure 31 – 20 nodes with 3 regions on the left and 4 on the right in the ENS scenario on E2

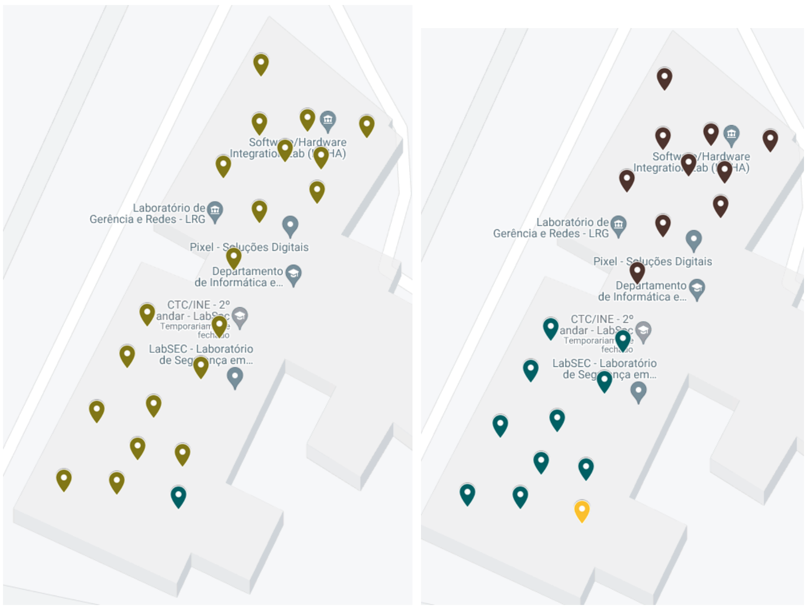

Software/Hardware Integration Lab (LIHA)
Laboratório de Gerência e Redes - LRG
Pixel - Soluções Digitais
Departamento de Informática e...
CTC/INE - 2º andar - LabSEC Temporariamente fechado
LabSEC - Laboratório de Segurança em...

Figure 32 – 20 nodes with 1 region on the left and 2 on the right in the INE E2 scenario

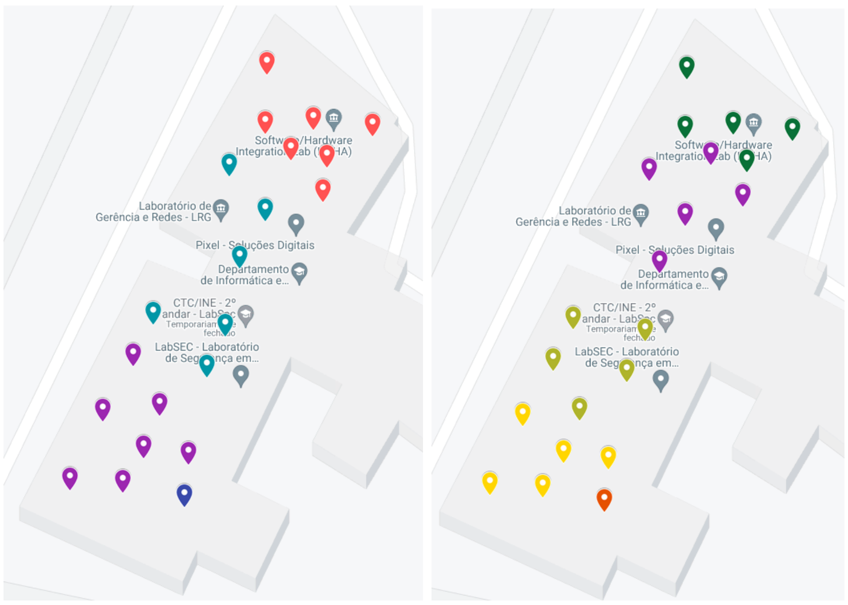

Software/Hardware Integration Lab (LIHA)
Laboratório de Gerência e Redes - LRG
Pixel - Soluções Digitais
Departamento de Informática e...
CTC/INE - 2º andar - LabSEC Temporariamente fechado
LabSEC - Laboratório de Segurança em...

Figure 33 – 20 nodes with 3 regions on the left and 4 on the right in the INE E2 scenario

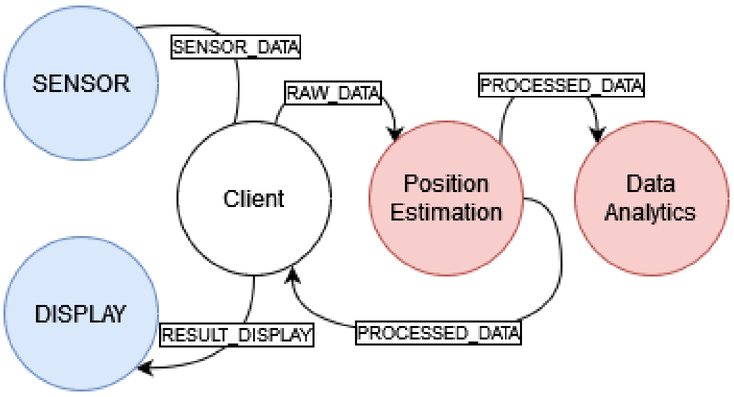


Figure 34 – Application Model used in OPS experiment

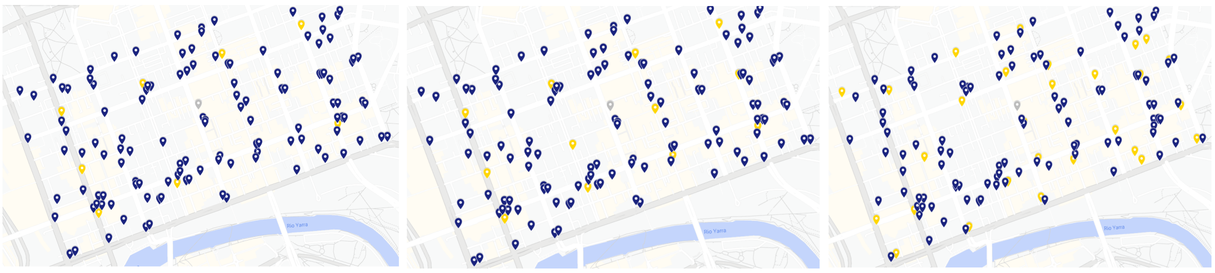


Figure 35 – Eight, twelve and twenty eight clusters configurations

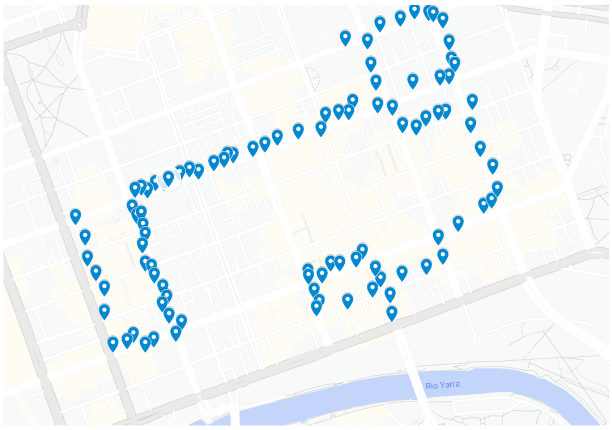


Figure 36 – A user path from south to east

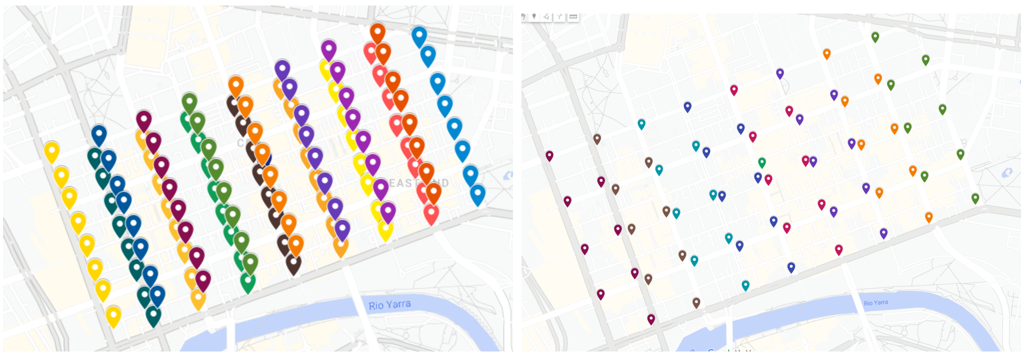


Figure 37 – Configurations for OPS environments (T1 on the left and T2 on the right)

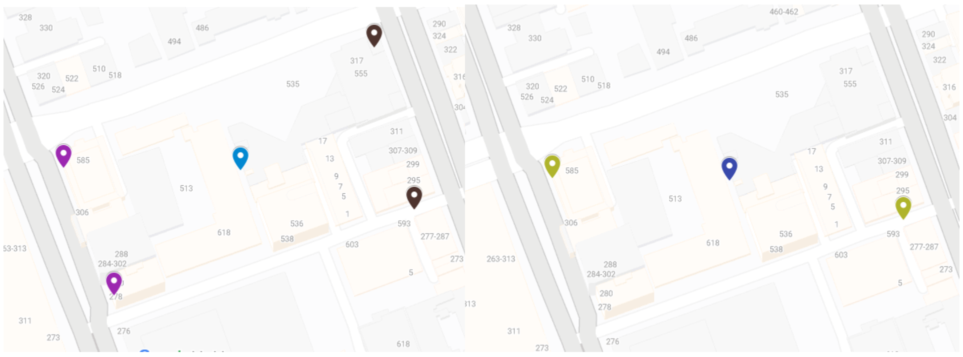


Figure 38 – Configurations for IPS environments (T1 on the left and T2 on the right)

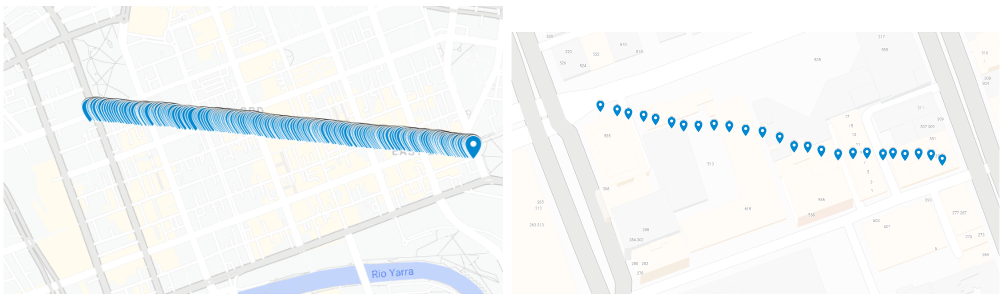


Figure 39 – Cross-area mobility pattern in outdoor (left) and indoor (right) environments

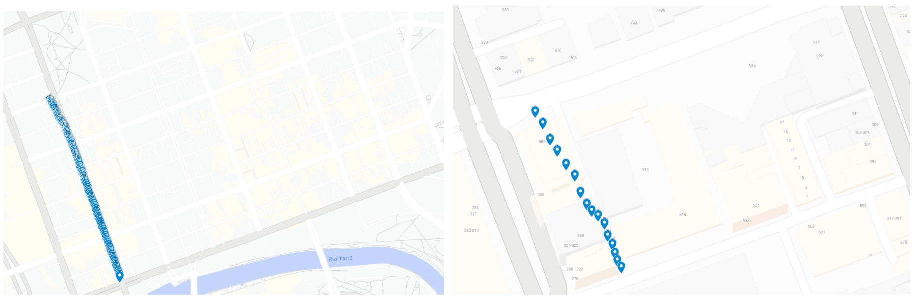


Figure 40 – Line-through mobility pattern in outdoor (left) and indoor (right) environments

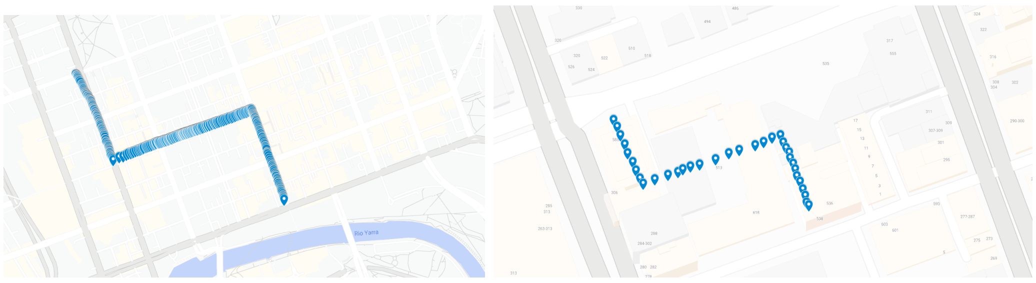


Figure 41 – Inline mobility pattern in indoor and outdoor environments

5 EXPERIMENTAL RESULTS AND EVALUATION

This section is dedicated to the presentation of experimental results from simulations and comparative analyses of approaches and configurations in different scenarios.

5.1 METRICS

Based on available metrics, the performance of each configuration was evaluated along with module placements using the mean and standard deviation. The average is calculated individually by the simulator at each round. The standard deviation was calculated from the 10 rounds in order to compare the stability of each configuration using the following data:

Average application delay time presents the overall performance; and how long it takes to complete the DAG flow. We use the mean of each setting. For this measure, the smaller the delay, the better the positioning and coordination decisions between computational resources. Values in milliseconds.

Migration time indicates the performance of the service migration, as it indicates the delay in the application modules. This metric is significantly more relevant for the clustered approach. This occurs because module transitions are more frequent in this approach, although they are not exclusive to it. Values in milliseconds.

Network usage indicates how much bandwidth the architecture used. Values in megabits per seconds.

5.2 EVALUATION OF INDOOR POSITIONING SYSTEMS EXPERIMENT

In this section, the results of each experiment are detailed for the IPS experiment referred to in subsection 4.7.1 based on the scenario, number of users, approach, and metrics.

5.2.1 Scenario ENS + E1

The subsections below present an analysis of the performance of the configurations presented in the subsection 4.7.1.3 in the ENS scenario, discussed in the subsection 4.7.1.2.

5.2.1.1 ALD benchmarks

In the ENS scenario with 5 users (Figure 42), the configuration with the best performance was clustered 12G-3P and the worst 4G-2P, also clustered. The allocation to the edges, on the other hand, remained the most stable and had the best average.The

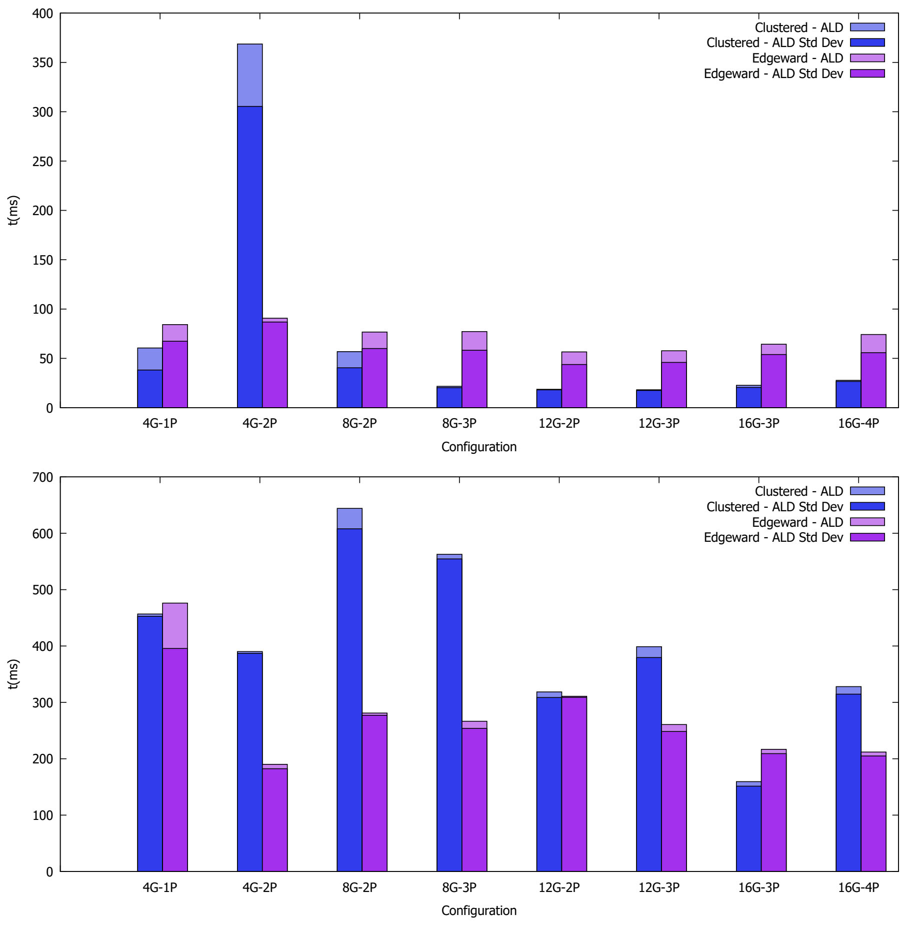


Figure 42 – ALD average per configuration in ENS with 5 and 25 users in E1

number of regions that stood out the most was the one with four, 51s, followed by three, with an average of 58.3s and the worst was 148.5s with 2. The average horizontal allocation by regions was 61s for a region , 148.3s for twos, 21s for threes and 28s for fours. As for the vertical, 84s for one, 75s for two, 66.3s for three and 74s for four. The general average, counting the standard deviations for the horizontal allocation was 74.6s, while for the vertical one, 72.75s.

With 25 users, the configuration with the best performance was 16G-3P with clustering and the worst, 8G-2P with the same allocation, as shown in Figure 42.Overall, the most stable was the edgewards, with a total average of 264.37s against the horizontal's 407.37s. In terms of regions, the average ranged from 466.5s for 1, 355.8s for two, 311s for three and 270 for four. For horizontal allocation, the average per region was 457s for one, 451s for two, 373.6s for three, and for four, 328s. While for vertical, 476s, 263.3s, 248.3s and 212s are observed for the regions in the same order.

5.2.1.2 NU benchmarks

Figure 43 presents the mean and standard deviation of network usage by configuration and allocation. This measure involves the data transferred between the layers

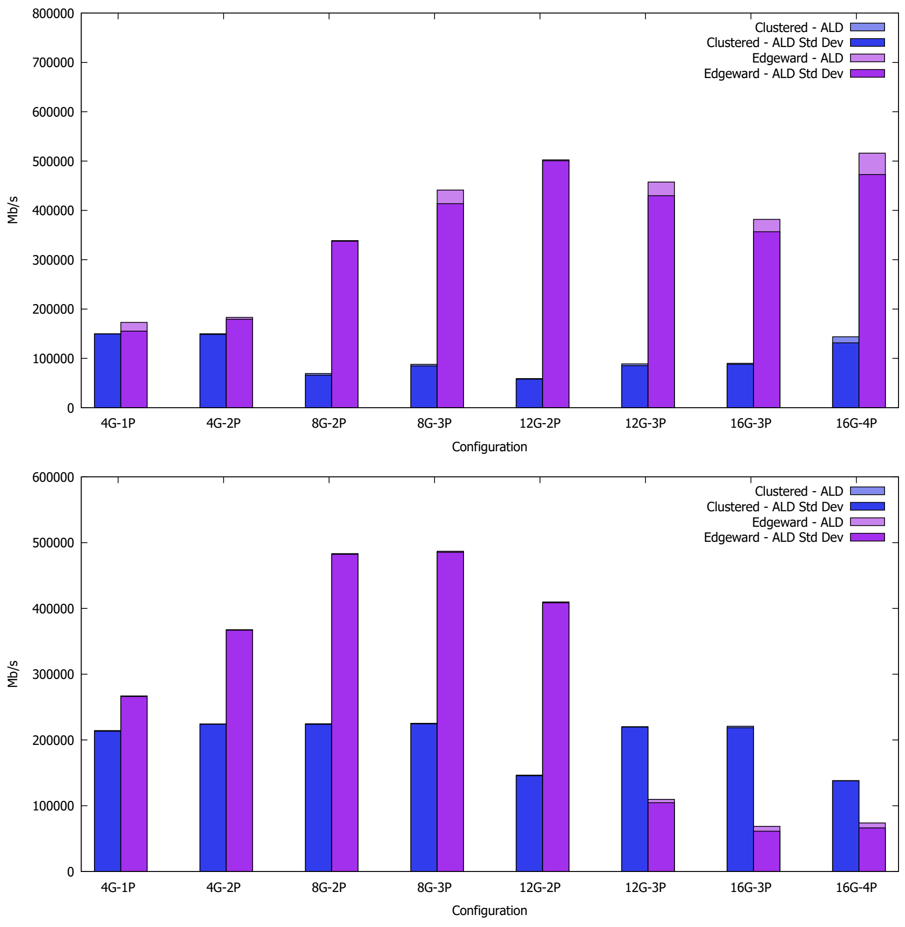


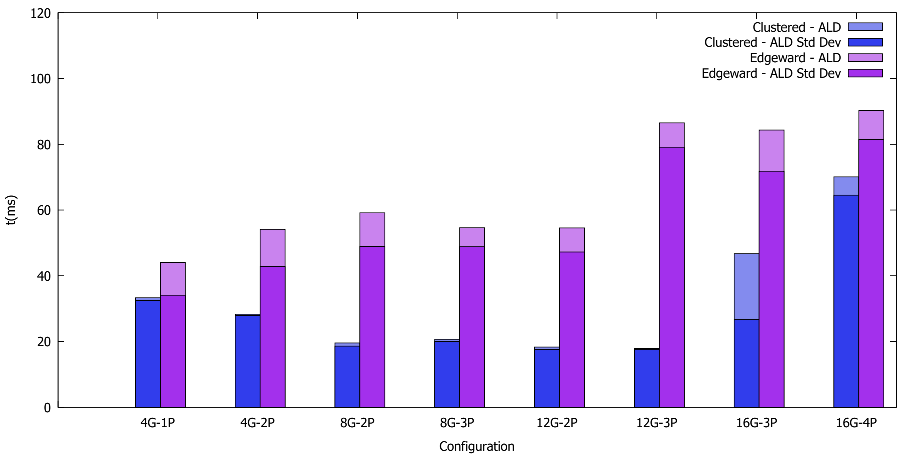
Figure 43 – Average NU per configuration in ENS with 5 and 25 users in E1

of the architecture. During the decision step, the proxy remains inactive, not consuming bandwidth. Except in two cases (4G-1P and 4G-2P), vertical allocation had higher bandwidth usage. The horizontal allocation remained stable in all cases, including its standard deviations, unlike the vertical one, which had standard deviations from an average of approximately 16%.

Low standard deviations indicate that network consumption did not vary considerably, as can be seen in both approaches, as is evident in Figure 43. However, in this scenario, vertical allocation used high bandwidth in most cases. Distinctly, with little variation, the vertical allocation remained stable.

5.2.2 Scenario INE + E1

The subsections below present an analysis of the performance of the configurations presented in the subsection 4.7.1.3 in the INE scenario, discussed in the subsection 4.7.1.2.



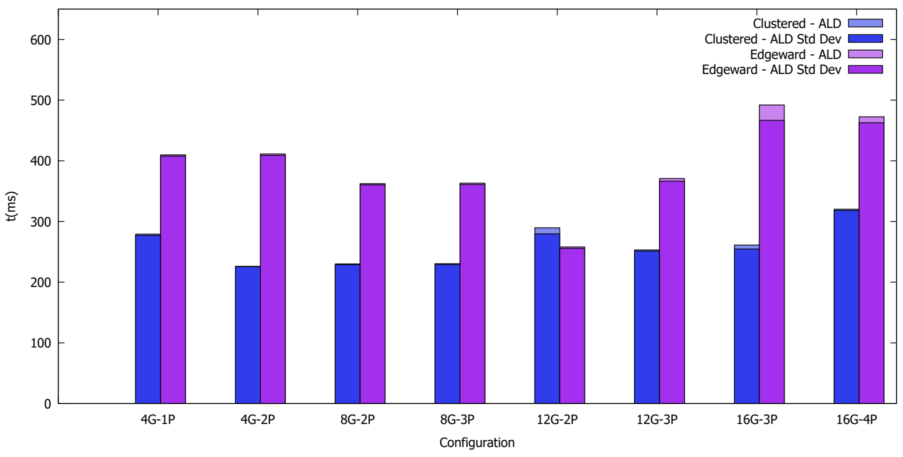


Figure 44 – Average ALD per configuration in INE with 5 (left) and 25 (right) users in E1

5.2.2.1 ALD benchmarks

According to Figure 44, for the scenario of 5 users, the clustered allocation had a lower completion time in all configurations, whereas the vertical allocation took longer to complete and had a higher standard deviation standard.As for the configurations, there was a tie for 12G-3P and 12G-2P in the clustered versions, while the worst, edgeward with 16G-4P.

The general average of the clustered approach was 31.6s, while the horizontal one was 66s. The regions had as averages: 38.5s for 1 region, 39s for 2, 52s for 3, and 80s for 4. While for regions by horizontal: 44s, 56s, 75.3 and 90, respectively. Finally, vertically: 33s, 21.3s, 28.6s, 70s for experiments with one to four regions.

In the experiment with 25 users, Figure 44, the vertical approach increased by 131s on average compared to the horizontal, with 261.125s for the first and 392.5s for the second. As for the regions in general, the average for each set of them was 344.5s, 296.1s, 328.3s and 396.5s. Clustered allocation averaged 279s, 248.6s, 248s, 320s for regions one through four, while vertical allocation averaged 410s, 343.6s, 408.6s, 473s.

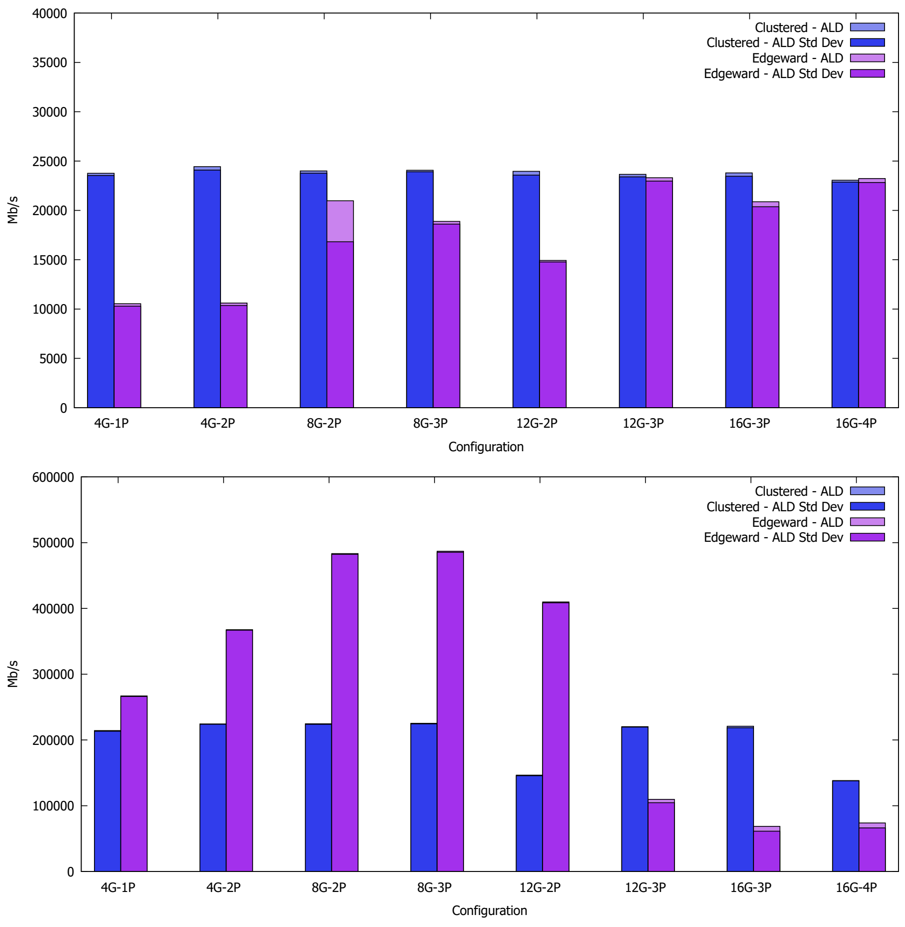


Figure 45 – Average NU per configuration in INE with 5 users in E1

5.2.2.2 NU benchmarks

Figure 45 presents the mean and standard deviation of network usage by configuration and allocation. This measure involves the data transferred between the layers of the architecture. During the decision step, the proxy remains inactive, not consuming bandwidth. Except in the case of 16G-4P, vertical allocation had lower bandwidth usage. The horizontal allocation remained stable in all cases, including its standard deviations.

As shown in Figure 45, the clustered approach used less bandwidth compared to the vertical one up to the 14 nodes (12 gateways and 2 proxies), although it remained stable. In addition, the standard deviations of network use were more expressive in the vertical approach.

5.2.3 Scenario ENS + E2

The subsections below present an analysis of the performance of the configurations presented in the subsection 4.7.1.4 in the INE scenario, discussed in the subsection 4.7.1.2.

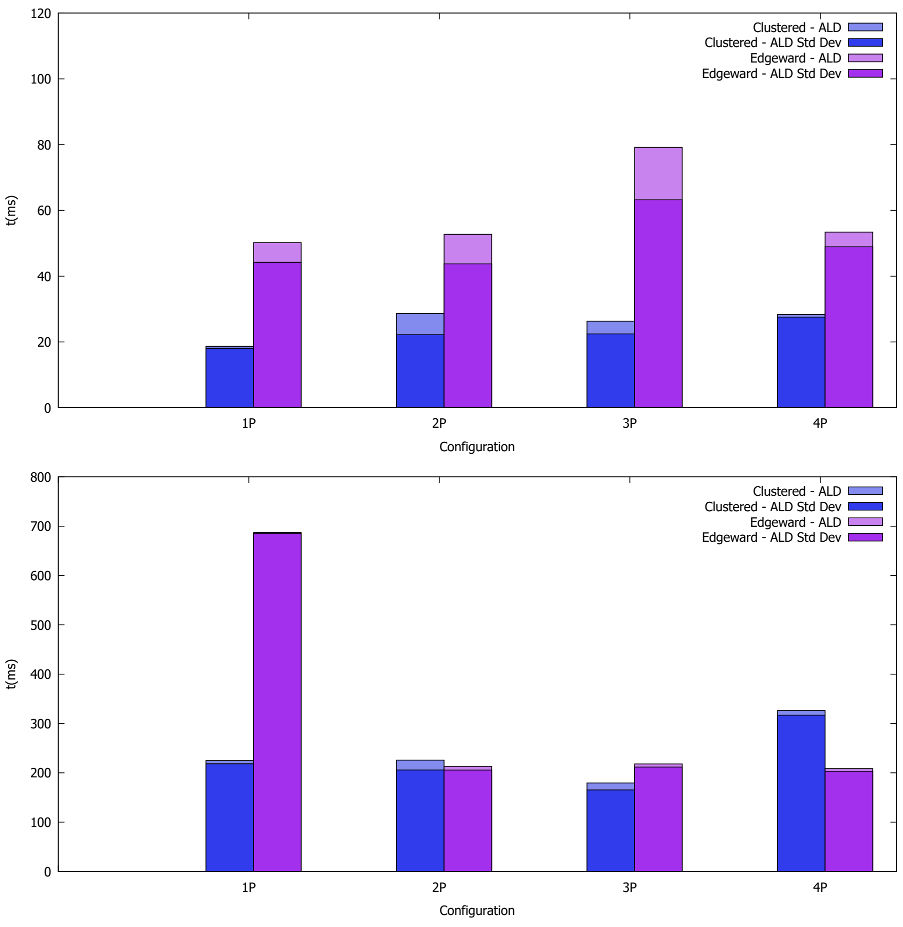


Figure 46 – Average ALD per configuration in ENS with 5 users on E2

5.2.3.1 ALD benchmarks

In the present case, the clustered approach performed better in all four cases, completing, on average for all cases, the application flow twice as fast as the vertical approach. It is also noted that the vertical one presented greater variation, as seen in Figure 46. The best result was driven by the clustered 1P approach and the worst, edgeward 3P.

In the scenarios with 25 users, the vertical approach had a greater range of performance for ALD, with the worst performance of the experiment with 687ms with 1P, as represented in Figure 46. In contrast, the horizontal approach varied less and stood out better with 3 regions.

5.2.3.2 NU benchmarks

In the scenario with 5 users (Figure 47), although the vertical approach had a greater variation in results, in all cases, it had lower bandwidth usage compared to the clustered approach. Horizontally, it can be seen that its stability is also maintained almost uniformly regardless of the number of regions.

With 25 users, the notable approach was the clustered one, which in all cases

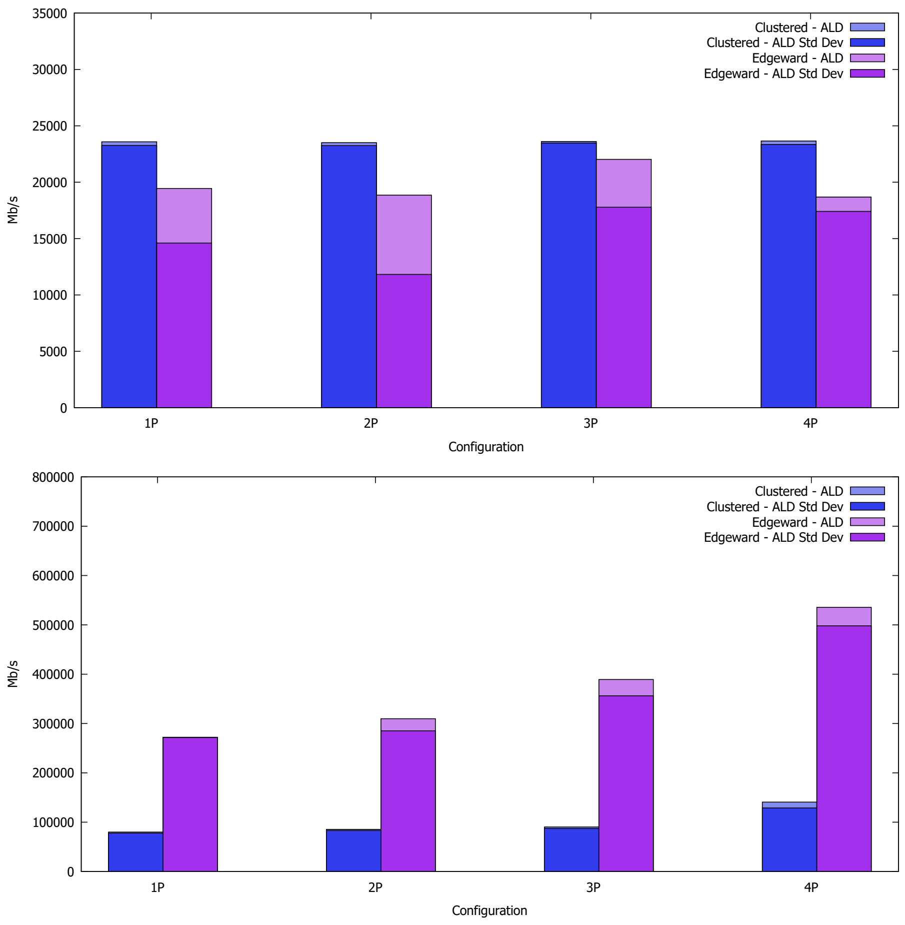


Figure 47 – Average of NU in ENS with 5 and 25 users on E2

used less than 30% of the network compared to the edgeward one, as shown in Figure 47. This approach had its lowest use with 1 region and the highest, 4 as well as the vertical approach.

5.2.4 Scenario INE + E2

The subsections below present an analysis of the performance of the configurations presented in the subsection 4.7.1.4 in the INE scenario, discussed in the subsection 4.7.1.2.

5.2.4.1 ALD benchmarks

Figure 48 shows the average ALD per configuration using 5 users. The strategy with the best performance was the clustered 1P and the worst, the vertical 3P, although the one with the greatest performance variation was the clustered 3P and 4P. It is possible to notice a direct correlation between the number of regions and application time, the greater the number of regions, the longer it takes to complete the application.

Figure 48 denotes the average application completion time per configuration with 25 users. It is observed that the configuration with the best performance was the

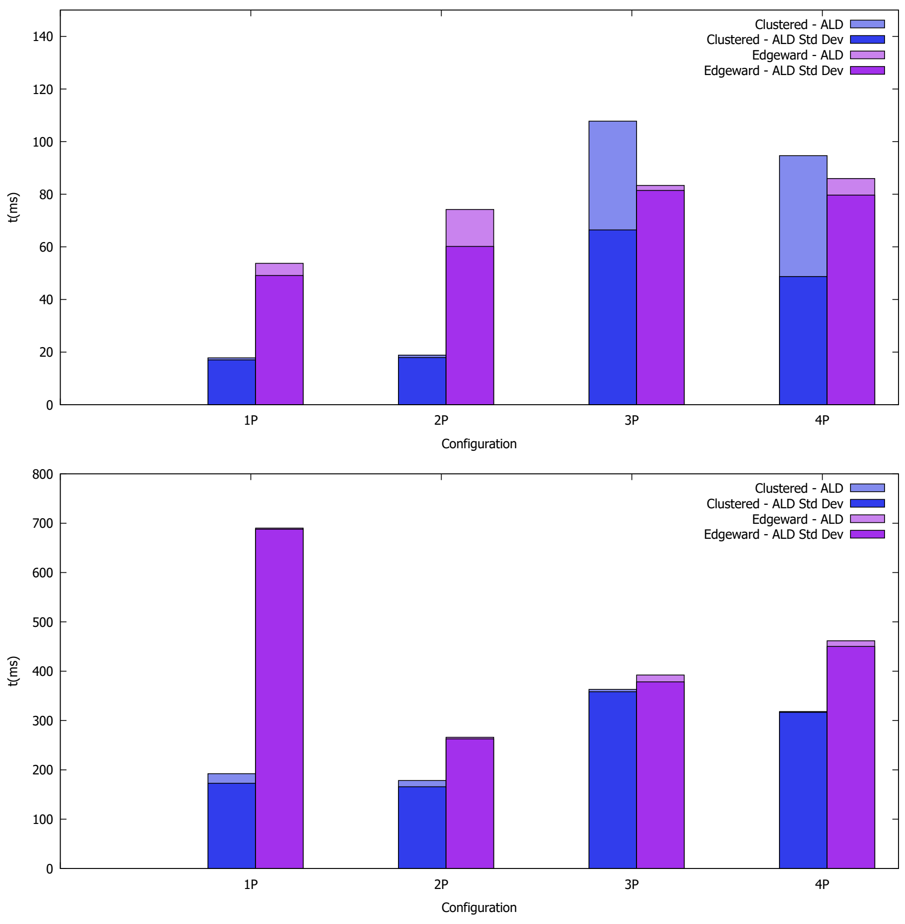


Figure 48 – Average ALD per configuration in INE with 5 and 25 users in E2

edgeward 2P while the worst 1P clustered. In this case, it appears that the relationship between the clustered approach and the number of proxies is 75% inversely proportional, while the vertical approach performed better with a smaller number of regions.

5.2.4.2 NU benchmarks

Although more stable, the clustered approach had higher network usage in all cases compared to the vertical approach, as shown in Figure 49. The configuration with the lowest network usage was the edgeward 1P versus the clustered 4P with the highest usage.

Despite having a configuration (4P) with higher network usage than the others, the horizontal approach had a notable lower bandwidth usage, especially the 2P with a use of 20% compared to the vertical approach. It is also noticed that the one with less variation between regions was the horizontal one, although there is an evident decline in network use between 2P and 3P for the vertical one.

In E1INE with 5 and 25 users, analyzing the ALD, it is possible to follow the same trend for both architectures in terms of regionality (12G-2P-3P). Although the proportionality of users is 5 times, both architectures had an increase of at least 10

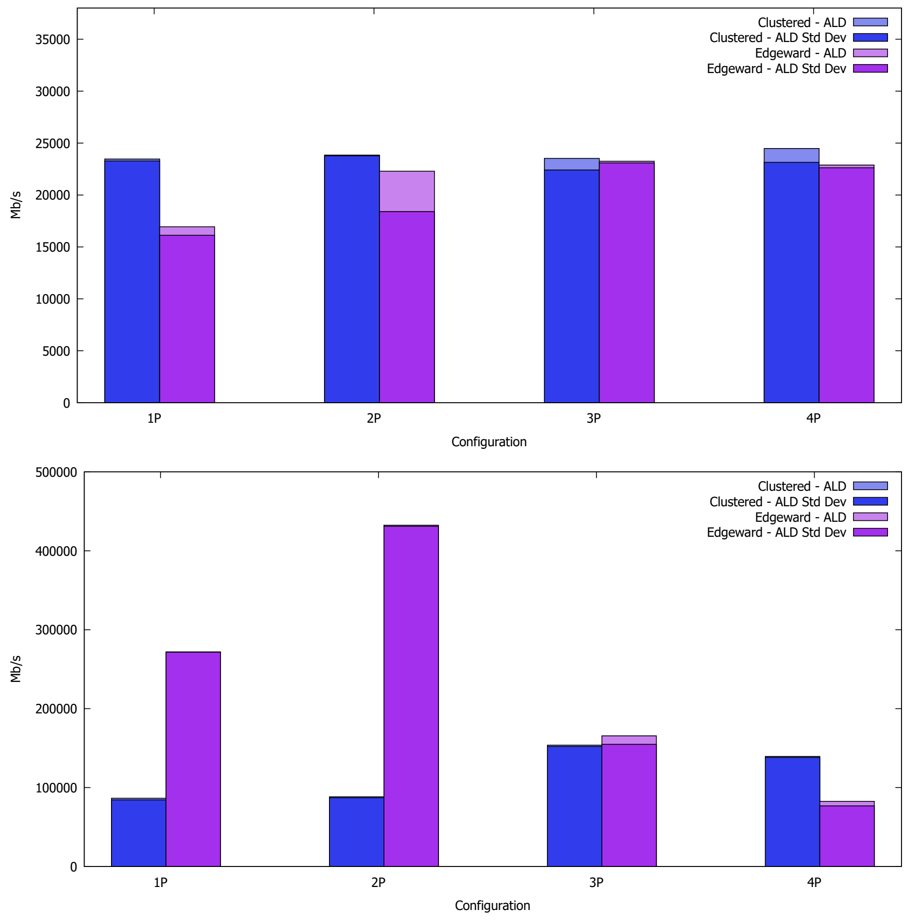


Figure 49 – Average of NU in INE with 5 and 25 users in E2

times.

5.2.5 Discussion

Experiment E1 sought to analyze how much the number of nodes can vary in the described metrics. In this, it was observed that the most successful approach (shorter execution time and lower bandwidth usage), in the environments presented, was the horizontal one in all cases. In E2, the same behavior was observed, however, in 75% of the two scenarios presented. Experiment E2 analyzed the impact of regionality for the same number of nodes. As for the relationship between regions and performance, it was found that architectures with fewer regions performed better for a few users, while for many users, there was a tendency for shorter execution times for multiple regions regardless of the approach used.

Analyzing the progression from 5 to 25 users individually for each configuration, there is an inversion of approaches in terms of performance in a 6/8 ratio, that is, of the 8 cases investigated, only 2 remained linearly equivalent. This fact denotes volatility in contemporary service allocation approaches, so that none of the proposed approaches scales linearly according to the number of users.

In general, the vertical approach resulted in higher standard deviations, confirming the greater stability of cluster-based solutions, though they may be less satisfactory.Even going into experiments E1 and E2 in ENS with 5 users, the high ALD in the horizontal approach is due to the positions of the regions, that is, because there are few gateways that cannot communicate, there is a task overhead in a region, culminating in a high *delay* at the completion of the application flow. It is also emphasized about the block location strategy, which caused great variation and inconclusive results in E1.

When compared to the horizontal approach, the vertical approach appears to be less suitable for limited networks, as it varies by more than 15% on average. Meanwhile, the low bandwidth standard deviations of the clustered scenarios indicated that network consumption did not vary considerably. In this way, the recurrent network stability in clustered systems, can be a positive aspect for the approximation of *space-application-agnostic* architectures, if they exist, as well as an auxiliary in SLA contracts, refining them. Even though the overall stability was higher, there were scenarios with fewer clients that had higher network use in comparison to the vertical.

The scenarios proposed in this work do not consider packet loss and obstacles such as walls or other aspects that lead to a decrease in performance. Although it can be understood as a limitation of the present work, it also highlights the challenge in the development and proof of the existence of totally *space application agnostic* architectures. Due to temporal linearity, all users in the experiments were active during the simulation. However, scenarios with network connection variability are important to evaluate the overall performance and are closer to reality.

5.3 EVALUATION OF OUTDOOR POSITIONING SYSTEMS EXPERIMENT

In this section, the results of each experiment are detailed for the OPS experiment referred to section 4.7.2 based on the scenario, number of users, approach and metrics. We used ALD and MT to evaluate the experiments.

5.3.1 Eight users scenario

In the eight users scenario, the ALD and NU increased proportionally to the number of clusters, and as a consequence, large clusters were more efficient. Figure 50 shows the ALD and NU and their standard deviations versus the number of clusters. Although the progression of the number of clusters increased both ALD and NU, the difference between clusters decreased. The ALD standard deviation, was an approximate 9% variation of the ALD and for NU, maximum 5%.

Similarly to the phenomenon effect in ALD and NU, the migration time increased accordingly to the number of clusters. Also, in Figure 51, it is possible to notice the difference decreasing when adding new clusters.

Figure 50 – Application loop delay and network usage per number of clusters in eight users scenario

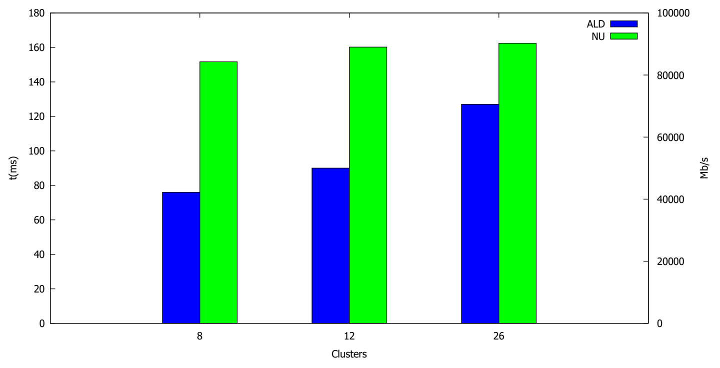
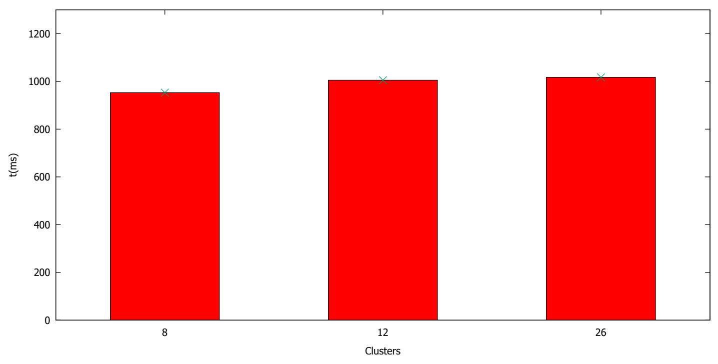


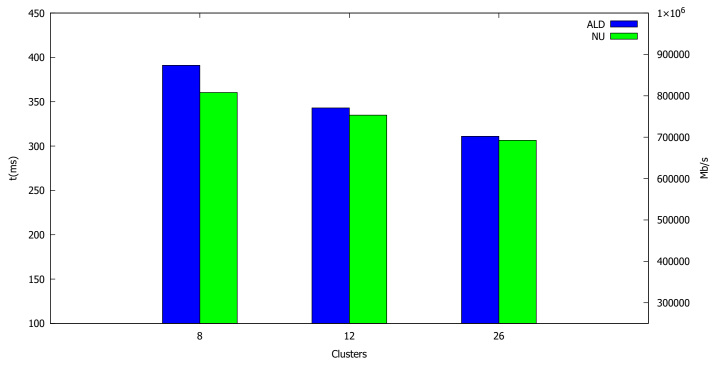
Figure 51 – Migration time per number of clusters in eight users scenario



5.3.2 Fifty users scenario

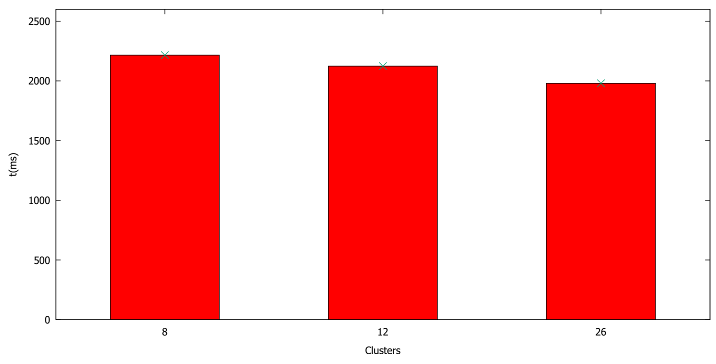
Figure 52 portrays that as the number of clusters increased, ALD, NU decreased, in contrast to the eight users scenario. For NU, as the service request-response was faster in smaller clusters, less bandwidth was used. Therefore, smaller clusters had a superior performance. As for the ALD standard deviation, even though the twelve clusters did not follow the progression, the mean was still proportional and varied up to 2%, while the NU, 1%.

Figure 52 – Application loop delay and network usage per cluster in the fifty users scenario



The figure 53 illustrates the negative relationship between the number of clusters and the MT. For the smaller clusters, this is a direct consequence of the application finishing faster.

Figure 53 – Migration time per number of clusters in fifty users scenario



5.3.3 Discussion

Using a small amount of users, the performance was better using larger (fewer) clusters. However, by adding multiple and simultaneous tasks from new users, the smaller clusters approach had a better outcome in all three metrics. Therefore, it was possible to demonstrate that there was no static configuration that works better in all user cases. Nonetheless, because the presented scenario is a metropolitan city block, smaller clusters can be used for better performance, as a small number of users is unlikely to be the norm.

5.4 EVALUATION OF OPS-IPS ENVIRONMENT

The results of each experiment for the IPS experiment referred to in section 4.7.3 are detailed in this section based on the scenario, number of users, approach, and metrics.In order to evaluate the configurations by comparing their performance to each mobility pattern and set of users, we selected the three metrics: ALD, NU and MT. Although indicators like cost in the cloud are equally important for the evaluation of the configurations, we selected the previously mentioned variables to reduce the scope of the study.

5.4.1 Indoor environment

In the indoor environment, the configuration T2 had better performance for 5 and 25 users when compared to the T1 configuration for all of the mobility cases, as presented in Figure 54. Interestingly, the performance was heavily equivalent for both sets of users in all three mobility patterns. For the configuration T1, the mobility pattern LT was the one that resulted in more time for completing the application loop for 5 and 25 users.

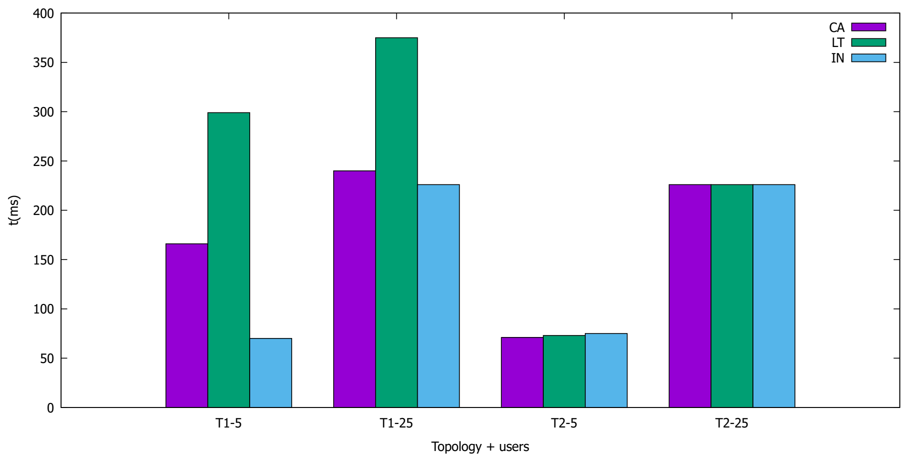


Figure 54 – Application Loop Delay for IPS based on mobility patterns

Figure 55 shows the migration time for the IPS. As can be seen, the amount of migration time for T2 was very low because all of the fog nodes belonged to the same cluster, and because the number of nodes' interactions was low, the simulator expressed a very low number for MT.The same thing happened to T1 in the IL for the same reason.However, LT for configuration T1 caused the most MT, having more than twice the value for CA. It is also important to note that the number of users was inversely proportional to the number of users for both mobility patterns, but more significantly for LT.

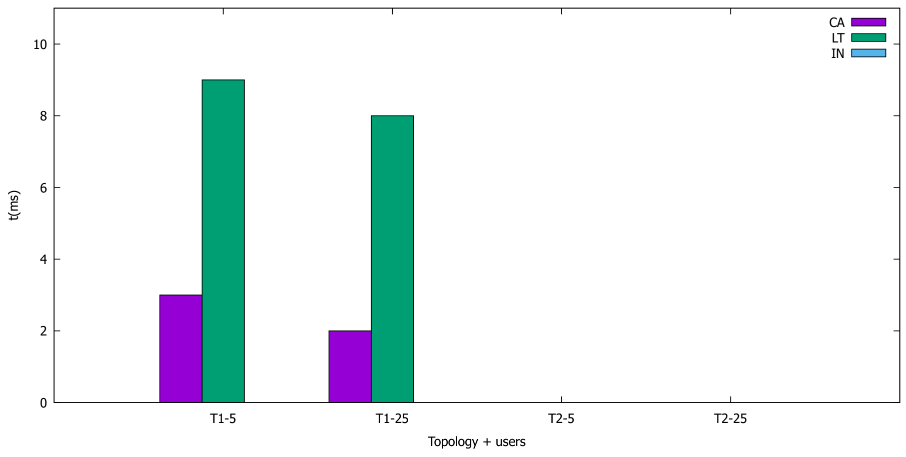


Figure 55 – IPS Migration Time based on mobility patterns

In terms of network usage, Figure 56 shows that the T1 used the least amount of bandwidth for 5 users, despite having the worst performance for NU with 25 users in LT.T2 was the most stable and used less network in all three MPs when compared to T1, with the exception of the LT in 5 users, which was less in T1.

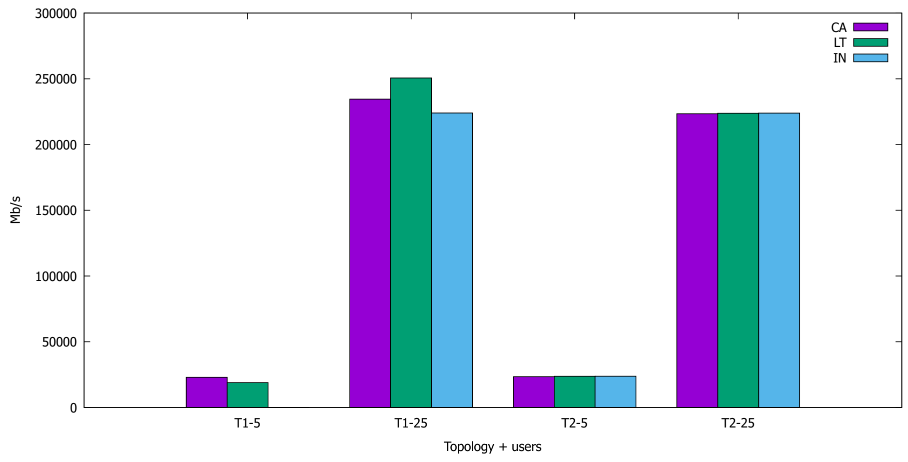


Figure 56 – IPS Network Usage based on mobility patterns

5.4.2 Outdoor environment

Analyzing the ALD for 5 and 25 users, the configurations were equivalent, as presented in Figure 57. The proportionality of the MP based on the number of users was followed in T1 and also in T2. As it is possible to notice, for 5 and 25 users, the MPs that used more time to get completed were CA, IL and then LT. The T2 showed better performance for all the mobility cases.

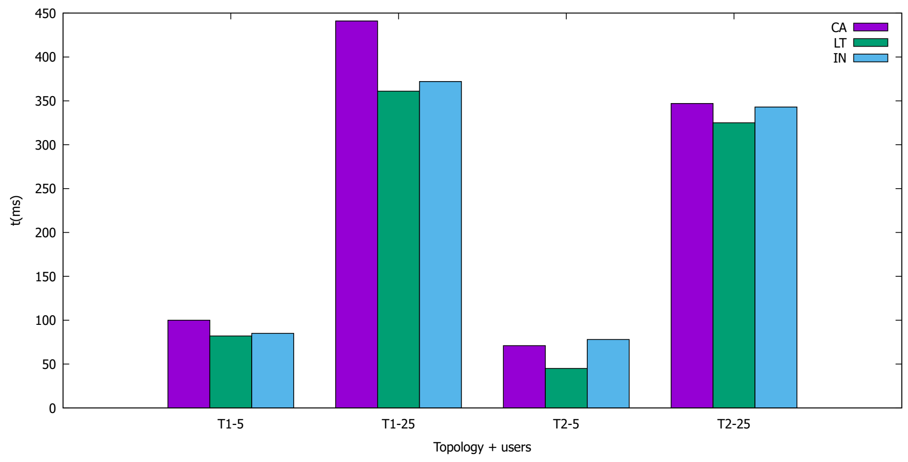


Figure 57 – Application Loop Delay for OPS based on mobility patterns

As shown in Figure 58, the configuration that used more time for the migration process for 5 users in CA, LT, and IL were T2, T1 and T2. The order of mobility patterns for 25 users was the same, indicating that there is a linear positive proportionality.For 5 and 25 users for each MP, the order from least to most used MT was T2, T1 and T2.

Figure 59 displays the NU for the OPS environment. The configuration that used less bandwidth for all of the mobility patterns independently of the set of users was T2. Linear positive proportionality is also possible to notice. For T2, the usage from 5 to 25 users was significantly less in comparison to T1.

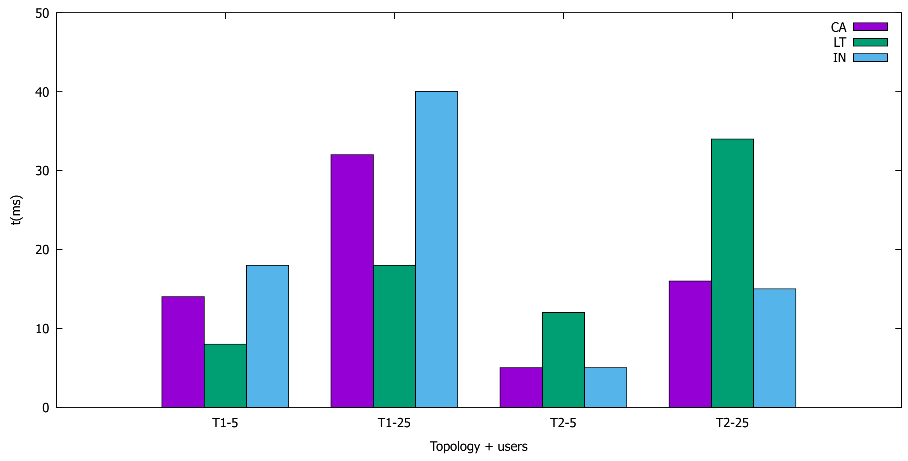


Figure 58 – Migration Time for OPS based on mobility patterns

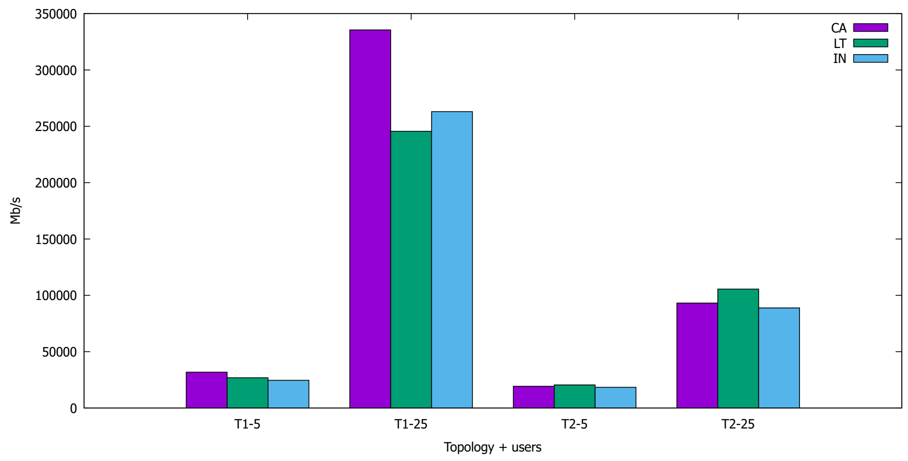


Figure 59 – Network Usage for OPS based on mobility patterns

5.4.3 Discussion

In OPS, it was possible to observe that there was a linear positive proportionality from 5 to 25 users for every mobility pattern in each configuration regarding the metrics (ALD, MT and MU). The configuration that was better for most of the cases was the T2. The T1 (MT, ALD) was superior for the LT mobility pattern because the nodes in the first cluster were capable of handling the tasks without needing to offload them to higher tiers or even wait.

The most demanding MP for IPS, was the LT for all of the presented metrics in T1. This happens because even though the nodes of the cluster were activated, the amount was not enough. As consequence, it was necessary to migrate intercluster, causing more delay for the completion. When there were fewer nodes and the option to migrate intercluster was removed, the performance was better and more even. Comparing the number of users to proposed metrics in any configuration and MP, these were directly proportional in OPS, but inversely proportional in IPS.

5.5 CHALLENGES AND LIMITATIONS

The number of users in the OPS environment was only 8 and 50 to understand how a base case behaves when adding new workloads. In IPS experiments, the objective was to study the machines' configuration and how the changes made to the fog nodes alter the overall performance of the ecosystem. Therefore, in the IPS experiments, the set of experimented users was smaller than the fog node configuration.

The experiments presented used fixed locations of nodes, so even though an experiment with 20 nodes was reduced to 10, we preserved the same location of previous nodes. It is important to emphasize that this location plays an important role in the system's performance. A possible feature could be a heatmap-based fog node location suggestion algorithm; as input, it could use an accepted node range, like the one presented in (MARTINS et al., 2022) and the user's interaction. This study could be possible using the code developed in this work [1].

[1] https://github.com/vyk1/service-placements-fog-mobility

6 CONCLUSION AND FUTURE WORK

In the present work, performance analyses of traditional service placements taking into account user mobility in location systems in indoor environments for fog architectures were proposed and developed through simulation. During the process, an outline of the state of the art in the open source community for fog and mobility was provided. With the available open source software, it was possible to identify performance metrics so that indoor experiments could be conducted.

The modifications and adaptations allowed the observed mobility to be analyzed through performance metrics. The modifications made to the iFogSim v2 simulator, extending it to understand internal scenarios, were also made available, enabling improvements and optimization for the academic community.

As the indoor environments presented significantly different behavior compared to the outdoor, an experimental procedure to inspect the traditional placements in OPS in contrast to IPS was developed. After different simulation experiments, the results suggest indoor and outdoor service placements may not be used interchangeably. In this manner, so does the concept of *space-application-agnosticism* in IPS environments, due to differences in numerous factors such as: position and number of regions, and network usage defined by the *Service Level Agreement* contracts.

Due to the variation of geographic spaces, movement, and number of users, it demonstrated the improbability of completely *space-agnostic* architectures using the traditional presented approaches. Likewise, a possible study *gap* is expressed about SLAs aimed at *indoor* location systems, culminating in possible contact breaks for architectures called *application agnostic*.

It was attested that the stability of the clusters can be a watershed aspect for the maintenance of the system's QoS, and that the migration time is one of the factors that can be better investigated for the development of more refined methodologies. As a result, it is worthwhile to consider developing a horizontal approach that is ready for indoor systems. This solution could be adaptive, as horizontal approaches perform better for larger numbers of users, such as adaptive clusters. This solution could benefit and deepen mobility and has not yet been explored. In addition, since indoor environments populated by people, like in museums and even smart airport contexts, the group-based mobility is a model that could be further investigated.

Improvement in the representation of distributed data models is another feature to be enhanced in order to reduce the migration time. As a possible solution, a hash table structure could facilitate task delegation using one of the mobility principles, such as approximation. However, it is worth mentioning within the ecosystem of *indoor* systems, that details prior to migration, such as the allocation and discovery of services, are state-of-the-art and should be explored within *indoor* systems, in this case, exclusively.

Besides, the importance of specialized mobile supported architectures is highlighted, since not every model handles any type of device interaction altogether, as it was possible to describe in the comparison study.

In spite of migrations, intra- and intercluster migrations can be the main issue related to the implacability of standard approaches on IPS. Nonetheless, a number of factors could be affecting traditional approaches performance in IPS. Firstly, the definition of the fog nodes location, since the WLAN nodes tend to be less powerful when compared to OPS machines and are outnumbered. Secondly, in the softwares considered, a maximum number of edge devices is allowed, requiring strategies to compare performance across environments.

As previously presented, IPS studies do not specify the dimensions of the tested environments. Therefore, there is a lack of definition for these environments: what are outdoor and indoor positioning systems? How far apart should they be? Is it possible to define both in meters per area? What about covered and uncovered ambients? In this case, the classification and definition of positioning environments are required for further research.

Still in the edge realm, mobility models for indoor environments should be more contextual. Currently, like the random waypoint described in section 2, there are variables like speed, direction, and randomness; however, surroundings are not taken into account. Also, most of the simulators are primarily context-agnostic, hindering the context-awareness needs of critical or sensible applications. Another challenge is defining realistic simulated environments, as the range of the nodes is not currently included in research values. This directly affects the reliability of the experiments, since in WiFi contexts the nodes' range varies.

Machine learning methods could be utilized to look into the optimal amount of clusters given a situation because the horizontal arrangement is nearer to the agnostic applications. With this strategy, it might be feasible to create the appropriate number of clusters, which would decrease idle machines and hence lower energy usage.

It might be able to develop more thorough situations using artificial intelligence. Machine learning algorithms could be used to define the user route depending on obstacles as well as in proactive migration. However, it is important to note that reliable input is required for accurate simulated surroundings. This means that a training dataset that is accurate to reality is essential for a successful orchestrations based on machine learning.

Since edge mobility tends to respect patterns, data extracted from location based systems could also enhance the system's performance. Microservice load balancers, for example, in clustered approaches, could proactively prepare machines to use a more specific amount of computing and energy power. And, besides possible IIoT environments, reactive methodologies could also use human mobility data to alleviate migration time.

REFERENCES

ADARSH, N; ARPITHA, J; ALI, Md Danish; CHARAN, N Mahesh; MAHENDRAKAR, Pramodini G. Effective blood bank management based on RFID in real time systems. In: IEEE. 2014 International Conference on Embedded Systems (ICES). [S.l.: s.n.], 2014. P. 287–290.

ADLER, Stephan; SCHMITT, Simon; WILL, Heiko; HILLEBRANDT, Thomas; KYAS, Marcel. Virtual testbed for indoor localization. In: INTERNATIONAL Conference on Indoor Positioning and Indoor Navigation. [S.l.: s.n.], 2013. P. 1–8. DOI: 10.1109/IPIN.2013.6817880.

AMIOT, Nicolas; LAARAIEDH, Mohamed; UGUEN, Bernard. PyLayers: An open source dynamic simulator for indoor propagation and localization. In: 2013 IEEE International Conference on Communications Workshops (ICC). [S.l.: s.n.], 2013. P. 84–88. DOI: 10.1109/ICCW.2013.6649206.

ANTONINI, Mattia; VECCHIO, Massimo; ANTONELLI, Fabio. Fog Computing Architectures: A Reference for Practitioners. **IEEE Internet of Things Magazine**, v. 2, n. 3, p. 19–25, 2019. DOI: 10.1109/IOTM.0001.1900029.

ANUP, S; GOEL, Abhinav; PADMANABHAN, Suresh. Visual positioning system for automated indoor/outdoor navigation. In: TENCON 2017 - 2017 IEEE Region 10 Conference. [S.l.: s.n.], 2017. P. 1027–1031. DOI: 10.1109/TENCON.2017.8228008.

ARIYAKHAJORN, Jinthana; WANNAWILAI, Pattana; SATHITWIRIYAWONG, Chanboon. A Comparative Study of Random Waypoint and Gauss-Markov Mobility Models in the Performance Evaluation of MANET. In: 2006 International Symposium on Communications and Information Technologies. [S.l.: s.n.], 2006. P. 894–899. DOI: 10.1109/ISCIT.2006.339866.

ASLANPOUR, Mohammad Sadegh; GILL, Sukhpal Singh; TOOSI, Adel. Performance Evaluation Metrics for Cloud, Fog and Edge Computing: A Review, Taxonomy, Benchmarks and Standards for Future Research. **Internet of Things**, v. 12, p. 100273, Aug. 2020. DOI: 10.1016/j.iot.2020.100273.

AWAISI, Kamran Sattar; ABBAS, Assad; KHAN, Samee U; MAHMUD, Redowan; BUYYA, Rajkumar. Simulating Fog Computing Applications using iFogSim Toolkit. **Mobile Edge Computing**, Springer, p. 565–590, 2021.

BAI, Fan; HELMY, Ahmed. A survey of mobility models. **Wireless Adhoc Networks. University of Southern California, USA**, v. 206, p. 147, 2004.

BATABYAL, Suvadip; BHAUMIK, Parama. Mobility Models, Traces and Impact of Mobility on Opportunistic Routing Algorithms: A Survey. **IEEE Communications Surveys & Tutorials**, v. 17, n. 3, p. 1679–1707, 2015. DOI: 10.1109/COMST.2015.2419819.

BATTISTONI, Pietro; SEBILLO, Monica; VITIELLO, Giuliana. Experimenting with a Fog-computing Architecture for Indoor Navigation. In: 2019 Fourth International Conference on Fog and Mobile Edge Computing (FMEC). [S.l.: s.n.], 2019. P. 161–165. DOI: 10.1109/FMEC.2019.8795307.

BAZO, Rodrigo; COSTA, Cristiano André da; SEEWALD, Lucas Adams; SILVEIRA, Luiz Gonzaga da; ANTUNES, Rodolfo Stoffel; RIGHI, Rodrigo da Rosa; RODRIGUES, Vinicius Facco. A Survey About Real-Time Location Systems in Healthcare Environments. **Journal of Medical Systems**, v. 45, n. 3, p. 35, Feb. 2021. ISSN 1573-689X. DOI: 10.1007/s10916-021-01710-1. Available from: https://doi.org/10.1007/s10916-021-01710-1.

BONOMI, Flavio; MILITO, Rodolfo; ZHU, Jiang; ADDEPALLI, Sateesh. Fog Computing and Its Role in the Internet of Things. In: PROCEEDINGS of the First Edition of the MCC Workshop on Mobile Cloud Computing. Helsinki, Finland: Association for Computing Machinery, 2012. (MCC ’12), p. 13–16. DOI: 10.1145/2342509.2342513. Available from: https://doi.org/10.1145/2342509.2342513.

CAMP, Tracy; BOLENG, Jeff; DAVIES, Vanessa. A survey of mobility models for ad hoc network research. **Wireless Communications and Mobile Computing**, v. 2, n. 5, p. 483–502, 2002. DOI: https://doi.org/10.1002/wcm.72. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/wcm.72. Available from: https://onlinelibrary.wiley.com/doi/abs/10.1002/wcm.72.

CISCO. **Fog Computing and the Internet of Things: Extend the Cloud to Where the Things Are**. 2015. (acessado em: 01.05.2021).

COUTINHO, Antonio; GREVE, Fabiola; PRAZERES, Cassio; CARDOSO, Joao. Fogbed: A Rapid-Prototyping Emulation Environment for Fog Computing. In: 2018 IEEE International Conference on Communications (ICC). [S.l.: s.n.], 2018. P. 1–7. DOI: 10.1109/ICC.2018.8423003.

COUTINHO, Antonio; RODRIGUES, Heitor; PRAZERES, Cássio; GREVE, Fabíola. Scalable Fogbed for Fog Computing Emulation. In: 2018 IEEE Symposium on Computers and Communications (ISCC). [S.l.: s.n.], 2018. P. 00334–00340. DOI: 10.1109/ISCC.2018.8538484.

FARAHSARI, Pooyan Shams; FARAHZADI, Amirhossein; REZAZADEH, Javad; BAGHERI, Alireza. A Survey on Indoor Positioning Systems for IoT-Based Applications. **IEEE Internet of Things Journal**, v. 9, n. 10, p. 7680–7699, 2022. DOI: 10.1109/JIOT.2022.3149048.

FAROOQUI, M. Najmul Islam; KHAN, Muhammad Mubashir; ARSHAD, Junaid; SHAFIQ, Omair. An empirical investigation of performance challenges within context-aware content sharing for vehicular ad hoc networks. **Transactions on Emerging Telecommunications Technologies**, v. 33, n. 10, e4157, 2022. DOI: https://doi.org/10.1002/ett.4157. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/ett.4157. Available from: https://onlinelibrary.wiley.com/doi/abs/10.1002/ett.4157.

FIROUZI, Farshad; FARAHANI, Bahar; WEINBERGER, Markus; DEPACE, Gabriel; ALIEE, Fereidoon Shams. IoT Fundamentals: Definitions, Architectures, Challenges, and Promises. In: **Intelligent Internet of Things: From Device to Fog and Cloud**. Ed. by Farshad Firouzi, Krishnendu Chakrabarty and Sani Nassif. Cham: Springer International Publishing, 2020. P. 3–50. ISBN 978-3-030-30367-9. DOI: 10.1007/978-3-030-30367-9_1. Available from: https://doi.org/10.1007/978-3-030-30367-9_1.

FORTIER, Paul J.; MICHEL, Howard E. 1 - Introduction. In: FORTIER, Paul J.; MICHEL, Howard E. (Eds.). **Computer Systems Performance Evaluation and Prediction**. Burlington: Digital Press, 2003. P. 1–38. ISBN 978-1-55558-260-9. DOI: https://doi.org/10.1016/B978-155558260-9/50001-1. Available from: https://www.sciencedirect.com/science/article/pii/B9781555582609500011.

GILL, Monika; SINGH, Dinesh. A comprehensive study of simulation frameworks and research directions in fog computing. **Computer Science Review**, v. 40, p. 100391, 2021. ISSN 1574-0137. DOI: `https://doi.org/10.1016/j.cosrev.2021.100391`. Available from: `https://www.sciencedirect.com/science/article/pii/S1574013721000319`.

GUBBI, Jayavardhana; BUYYA, Rajkumar; MARUSIC, Slaven; PALANISWAMI, Marimuthu. Internet of Things (IoT): A vision, architectural elements, and future directions. **Future Generation Computer Systems**, v. 29, n. 7, p. 1645–1660, 2013. Including Special sections: Cyber-enabled Distributed Computing for Ubiquitous Cloud and Network Services & Cloud Computing and Scientific Applications — Big Data, Scalable Analytics, and Beyond. ISSN 0167-739X. DOI: `https://doi.org/10.1016/j.future.2013.01.010`. Available from: `https://www.sciencedirect.com/science/article/pii/S0167739X13000241`.

GUO, Yanjun; ZHAO, Liqiang; WANG, Yong; LIU, Qi; QIU, Jiahui. Fog-Enabled WLANs for Indoor Positioning. In: 2019 IEEE 89th Vehicular Technology Conference (VTC2019-Spring). [S.l.: s.n.], 2019. P. 1–5. DOI: `10.1109/VTCSpring.2019.8746592`.

GUPTA, Harshit; VAHID DASTJERDI, Amir; GHOSH, Soumya K; BUYYA, Rajkumar. iFogSim: A toolkit for modeling and simulation of resource management techniques in the Internet of Things, Edge and Fog computing environments. **Software: Practice and Experience**, Wiley Online Library, v. 47, n. 9, p. 1275–1296, 2017.

HUANG, ChingYao et al. Trusted worthy Fog Computing testbed developed in Great China Region. In: 2017 IEEE Fog World Congress (FWC). [S.l.: s.n.], 2017. P. 1–6. DOI: `10.1109/FWC.2017.8368536`.

IORGA, Michaela; FELDMAN, Larry; BARTON, Robert; MARTIN, Michael; GOREN, Nedim; MAHMOUDI, Charif. **Fog Computing Conceptual Model**. en. [S.l.]: Special Publication (NIST SP), National Institute of Standards and Technology, Gaithersburg, MD, 2018. DOI: `https://doi.org/10.6028/NIST.SP.500-325`.

JACHIMCZYK, Bartosz. **Real-time Locating Systems for indoor applications : the methodological customization approach**. 2019. S. 204. PhD thesis – DAC SA. ISBN 978-91-7295-373-4.

JANKOWSKI, Tomasz; NIKODEM, Maciej. SMILe - Simulator for Methods of Indoor Localization. In: 2018 International Conference on Indoor Positioning and Indoor Navigation (IPIN). [S.l.: s.n.], 2018. P. 206–212. DOI: 10.1109/IPIN.2018.8533754.

JOHN, Jobish; GHOSAL, Amrita; MARGARIA, Tiziana; PESCH, Dirk. DSLs for Model Driven Development of Secure Interoperable Automation Systems with EdgeX Foundry. In: 2021 Forum on specification & Design Languages (FDL). [S.l.: s.n.], 2021. P. 1–8. DOI: 10.1109/FDL53530.2021.9568378.

KARWA, Radhika. Realtime Indoor Location-Based Passenger Tracking System using Bluetooth Beacon for Airport Authority. **International Journal for Research in Applied Science and Engineering Technology**, v. 7, p. 3617–3626, Apr. 2019. DOI: 10.22214/ijraset.2019.4607.

KUNDE, Christian; MANN, Zoltán Ádám. Comparison of Simulators for Fog Computing. In: PROCEEDINGS of the 35th Annual ACM Symposium on Applied Computing. Brno, Czech Republic: Association for Computing Machinery, 2020. (SAC ’20), p. 1792–1795. DOI: 10.1145/3341105.3375771. Available from: https://doi.org/10.1145/3341105.3375771.

LAI, Phu; HE, Qiang; ABDELRAZEK, Mohamed; CHEN, Feifei; HOSKING, John; GRUNDY, John; YANG, Yun. Optimal Edge User Allocation in Edge Computing with Variable Sized Vector Bin Packing. In: PAHL, Claus; VUKOVIC, Maja; YIN, Jianwei; YU, Qi (Eds.). **Service-Oriented Computing**. Cham: Springer International Publishing, 2018. P. 230–245.

LAN, Dapeng; TAHERKORDI, Amir; ELIASSEN, Frank; HORN, Geir. A Survey on Fog Programming: Concepts, State-of-the-Art, and Research Challenges. In: PROCEEDINGS of the 2nd International Workshop on Distributed Fog Services Design. Davis, CA, USA: Association for Computing Machinery, 2019. (DFSD ’19), p. 1–6. DOI: 10.1145/3366613.3368120. Available from: https://doi.org/10.1145/3366613.3368120.

LEHRIG, Sebastian; EIKERLING, Hendrik; BECKER, Steffen. Scalability, elasticity, and efficiency in cloud computing: A systematic literature review of definitions and metrics. In: 2015 11th International ACM SIGSOFT Conference on Quality of Software Architectures (QoSA). [S.l.: s.n.], 2015. P. 83–92. DOI: 10.1145/2737182.2737185.

LERA, Isaac; GUERRERO, Carlos; JUIZ, Carlos. YAFS: A simulator for IoT scenarios in fog computing. **IEEE Access**, IEEE, v. 7, p. 91745–91758, 2019.

LOPES, Márcio Moraes; HIGASHINO, Wilson A; CAPRETZ, Miriam AM; BITTENCOURT, Luiz Fernando. Myifogsim: A simulator for virtual machine migration in fog computing. In: COMPANION Proceedings of the10th International Conference on Utility and Cloud Computing. [S.l.: s.n.], 2017. P. 47–52.

MAAD HAMDI, Mustafa; AUDAH, Lukman; ABDULJABBAR RASHID, Sami; HAMID MOHAMMED, Alaa; ALANI, Sameer; SHAMIL MUSTAFA, Ahmed. A Review of Applications, Characteristics and Challenges in Vehicular Ad Hoc Networks (VANETs). In: 2020 International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA). [S.l.: s.n.], 2020. P. 1–7. DOI: `10.1109/HORA49412.2020.9152928`.

MACEDO, Douglas DJ de; VON WANGENHEIM, Aldo; DANTAS, Mario AR. A data storage approach for large-scale distributed medical systems. In: IEEE. 2015 Ninth International Conference on Complex, Intelligent, and Software Intensive Systems. [S.l.: s.n.], 2015. P. 486–490.

MACEDO, Douglas Dyllon Jeronimo de; ARAÚJO, Gustavo Medeiros de; DUTRA, Moisés Lima; DUTRA, Silvana Toriani; LEZANA, Álvaro Guillermo Rojas. Toward an efficient healthcare CloudIoT architecture by using a game theory approach. **Concurrent Engineering**, v. 27, n. 3, p. 189–200, 2019. DOI: `10.1177/1063293X19844548`. eprint: `https://doi.org/10.1177/1063293X19844548`. Available from: `https://doi.org/10.1177/1063293X19844548`.

MACKEY, Andrew; SPACHOS, Petros. Performance evaluation of beacons for indoor localization in smart buildings. In: 2017 IEEE Global Conference on Signal and Information Processing (GlobalSIP). [S.l.: s.n.], 2017. P. 823–827. DOI: `10.1109/GlobalSIP.2017.8309075`.

MAHMUD, Redowan; PALLEWATTA, Samodha; GOUDARZI, Mohammad; BUYYA, Rajkumar. iFogSim2: An extended iFogSim simulator for mobility, clustering, and microservice management in edge and fog computing environments. **Journal of Systems and Software**, Elsevier, v. 190, p. 111351, 2022.

MAINETTI, Luca; PATRONO, Luigi; SERGI, Ilaria. A survey on indoor positioning systems. In: 2014 22nd International Conference on Software, Telecommunications and Computer Networks (SoftCOM). [S.l.: s.n.], 2014. P. 111–120. DOI: 10.1109/SOFTCOM.2014.7039067.

MANSOURI, Yaser; BABAR, M. Ali. A review of edge computing: Features and resource virtualization. **Journal of Parallel and Distributed Computing**, v. 150, p. 155–183, 2021. ISSN 0743-7315. DOI: https://doi.org/10.1016/j.jpdc.2020.12.015. Available from: https://www.sciencedirect.com/science/article/pii/S0743731520304317.

MARCONI, Marina de Andrade; LAKATOS, Eva Maria. Técnicas de pesquisa: planejamento e execução de pesquisa; amostragens e técnicas de pesquisa; elaboração, análise e interpretação de dados. In: TÉCNICAS de pesquisa: planejamento e execução de pesquisa; amostragens e técnicas de pesquisa; elaboração, análise e interpretação de dados. [S.l.: s.n.], 2012. P. 277–277.

MARGARITI, S.; DIMAKOPOULOS, Vassilios; TSOUMANIS, Georgios. Modeling and Simulation Tools for Fog Computing-A Comprehensive Survey from a Cost Perspective. **Future Internet**, v. 12, p. 89, May 2020. DOI: 10.3390/fi12050089.

MARIN, Iuliana; BOCICOR, Maria; MOLNAR, Arthur-Jozsef. Intelligent Luminaire based Real-time Indoor Positioning for Assisted Living. **Proceedings of the 15th International Conference on Evaluation of Novel Approaches to Software Engineering**, SCITEPRESS - Science and Technology Publications, 2020. DOI: 10.5220/0009578705480555. Available from: http://dx.doi.org/10.5220/0009578705480555.

MARTINS, Victoria B; MACEDO, Douglas DJ de; PIOLI JR, Laércio; IMMICH, Roger. A Cluster Formation Algorithm for Fog Architectures Based on Mobility Parameters at a Geographically LAN Perspective. In: SPRINGER. ADVANCES on P2P, Parallel, Grid, Cloud and Internet Computing: Proceedings of the 17th International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC-2022). [S.l.: s.n.], 2022. P. 25–36.

MAYER, Ruben; GRASER, Leon; GUPTA, Harshit; SAUREZ, Enrique; RAMACHANDRAN, Umakishore. EmuFog: Extensible and scalable emulation of large-scale fog computing infrastructures. In: 2017 IEEE Fog World Congress (FWC). [S.l.: s.n.], 2017. P. 1–6. DOI: 10.1109/FWC.2017.8368525.

NASIR, Mansoor; MUHAMMAD, Khan; LLORET, Jaime; SANGAIAH, Arun Kumar; SAJJAD, Muhammad. Fog computing enabled cost-effective distributed summarization of surveillance videos for smart cities. **Journal of Parallel and Distributed Computing**, v. 126, p. 161–170, 2019. ISSN 0743-7315. DOI: `https://doi.org/10.1016/j.jpdc.2018.11.004`. Available from: `https://www.sciencedirect.com/science/article/pii/S0743731518308402`.

OCAWG. **OpenFog Reference Architecture for Fog Computing**. 2017. (acessado em: 18.08.2021).

OROYA-VILLALTA, Deyviss; DÍEZ, Luis; BAHILLO, Alfonso. Navindoor: Plataforma de simulación para el diseño, prueba y evaluación de sistemas de localización. In.

PALLEWATTA, Samodha; KOSTAKOS, Vassilis; BUYYA, Rajkumar. Microservices-Based IoT Application Placement within Heterogeneous and Resource Constrained Fog Computing Environments. In: PROCEEDINGS of the 12th IEEE/ACM International Conference on Utility and Cloud Computing. Auckland, New Zealand: Association for Computing Machinery, 2019. (UCC'19), p. 71–81. DOI: `10.1145/3344341.3368800`. Available from: `https://doi.org/10.1145/3344341.3368800`.

PEŠIĆ, Saša; TOŠIĆ, Milenko; IKOVIĆ, Ognjen; RADOVANOVIĆ, Miloš; IVANOVIĆ, Mirjana; BOŠKOVIĆ, Dragan. BLEMAT: Data Analytics and Machine Learning for Smart Building Occupancy Detection and Prediction. **International Journal on Artificial Intelligence Tools**, v. 28, p. 1960005, Sept. 2019. DOI: `10.1142/S0218213019600054`.

PEŠIĆ, Saša; TOŠIĆ, Milenko; IKOVIĆ, Ognjen; RADOVANOVIĆ, Miloš; IVANOVIĆ, Mirjana; BOŠKOVIĆ, Dragan. Bluetooth Low Energy Microlocation Asset Tracking (BLEMAT) in a Context-Aware Fog Computing System. In: (WIMS '18). DOI: `10.1145/3227609.3227652`. Available from: `https://doi.org/10.1145/3227609.3227652`.

PHUNG, James; YOUNG, Choon Lee; ZOMAYA, Albert Y. Application-Agnostic Power Monitoring in Virtualized Environments. In: 2017 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID). [S.l.: s.n.], 2017. P. 335–344. DOI: `10.1109/CCGRID.2017.100`.

PIETRABISSA, Antonio; POLI, Cecilia; FERRIERO, Dario Giuseppe; GRIGIONI, Mauro. Optimal planning of sensor networks for asset tracking in hospital environments. **Decision Support Systems**, v. 55, n. 1, p. 304–313, 2013. ISSN 0167-9236. DOI: `https://doi.org/10.1016/j.dss.2013.01.031`. Available from: `https://www.sciencedirect.com/science/article/pii/S0167923613000596`.

PIOLI, Laércio; DORNELES, Carina F.; MACEDO, Douglas D. J. de; DANTAS, Mario A. R. An overview of data reduction solutions at the edge of IoT systems: a systematic mapping of the literature. **Computing**, v. 104, n. 8, p. 1867–1889, 2022. ISSN 1436-5057. DOI: `10.1007/s00607-022-01073-6`. Available from: `https://doi.org/10.1007/s00607-022-01073-6`.

PSOMAKELIS, Evangelos; TSERPES, Konstantinos; ZISSIS, Dimitris; ANAGNOSTOPOULOS, Dimosthenis; VARVARIGOU, Theodora. Context agnostic trajectory prediction based on λ-architecture. **Future Generation Computer Systems**, v. 110, p. 531–539, 2020. ISSN 0167-739X. DOI: `https://doi.org/10.1016/j.future.2019.09.046`. Available from: `https://www.sciencedirect.com/science/article/pii/S0167739X17320204`.

SAMANN, Fady Esmat Fathel; ZEEBAREE, Subhi RM; ASKAR, Shavan. IoT provisioning QoS based on cloud and fog computing. **Journal of Applied Science and Technology Trends**, v. 2, n. 01, p. 29–40, 2021.

SANTOS, Alessandro; AVANÇO, Leandro; PEREIRA, Matheus. **TECNOLOGIAS EMERGENTES EM IOT: RSSF, RTLS, RFID CONCEITOS E APLICAÇÕES PARA CIDADES INTELIGENTES E INDÚSTRIA 4.0**. [S.l.: s.n.], Jan. 2020. ISBN 9786557020005.

SCHMITT, Simon; WILL, Heiko; ASCHENBRENNER, Benjamin; HILLEBRANDT, Thomas; KYAS, Marcel. A reference system for indoor localization testbeds. In: 2012 International Conference on Indoor Positioning and Indoor Navigation (IPIN). [S.l.: s.n.], 2012. P. 1–8. DOI: `10.1109/IPIN.2012.6418865`.

SILVA, Daniel Maniglia A. da; ASAAMONING, Godwin; ORRILLO, Hector; SOFIA, Rute C.; MENDES, Paulo M. An Analysis of Fog Computing Data Placement Algorithms. In: PROCEEDINGS of the 16th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services. Houston, Texas, USA: Association for Computing Machinery, 2019. (MobiQuitous ’19), p. 527–534. DOI:

10.1145/3360774.3368201. Available from:
https://doi.org/10.1145/3360774.3368201.

SILVA, Heitor Rodrigues Santos; COUTINHO, Antonio. A Distributed Environment for Scalable Fog Computing Emulation. **Engenharia de Computação da UEFS**, p. 1–10, 2018.

SOKOLOVA, Nadezda. **Navigation, positioning and localization**. Available from: https://www.sintef.no/en/expertise/information-and-communication-technology-ict/communication-systems/navigation/. (acessado em: 12.07.2021).

SONMEZ, Cagatay; OZGOVDE, Atay; ERSOY, Cem. Edgecloudsim: An environment for performance evaluation of edge computing systems. **Transactions on Emerging Telecommunications Technologies**, Wiley Online Library, v. 29, n. 11, e3493, 2018.

SPACHOS, Petros; PLATANIOTIS, Konstantinos N. BLE Beacons for Indoor Positioning at an Interactive IoT-Based Smart Museum. **IEEE Systems Journal**, v. 14, n. 3, p. 3483–3493, 2020. DOI: 10.1109/JSYST.2020.2969088.

SYAFRUDIN, Muhammad; LEE, Keeeun; ALFIAN, Ganjar; LEE, Jaeho; RHEE, Jongtae. Application of Bluetooth Low Energy-Based Real-Time Location System for Indoor Environments. In: PROCEEDINGS of the 2018 2nd International Conference on Big Data and Internet of Things. Beijing, China: Association for Computing Machinery, 2018. (BDIOT 2018), p. 167–171. DOI: 10.1145/3289430.3289470. Available from: https://doi.org/10.1145/3289430.3289470.

SYMEONIDES, Moysis; GEORGIOU, Zacharias; TRIHINAS, Demetris; PALLIS, George; DIKAIAKOS, Marios D. Fogify: A Fog Computing Emulation Framework. In: 2020 IEEE/ACM Symposium on Edge Computing (SEC). [S.l.: s.n.], 2020. P. 42–54. DOI: 10.1109/SEC50012.2020.00011.

TRAN, Minh Quang; NGUYEN, Phat; TSUCHIYA, Takeshi; TOULOUSE, Michel. Designed Features for Improving Openness, Scalability and Programmability in the Fog Computing-Based IoT Systems. **SN Computer Science**, v. 1, June 2020. DOI: 10.1007/s42979-020-00197-w.

VANINI, Salvatore; GIORDANO, Silvia. Adaptive Context-Agnostic Floor Transition Detection on Smart Mobile Devices. In: p. 2–7. DOI: 10.1109/PerComW.2013.6529447.

VARSHNEY, Vibhu; GOEL, Rajat Kant; QADEER, Mohammed Abdul. Indoor positioning system using Wi-Fi amp; Bluetooth Low Energy technology. In: 2016 Thirteenth International Conference on Wireless and Optical Communications Networks (WOCN). [S.l.: s.n.], 2016. P. 1–6. DOI: 10.1109/WOCN.2016.7759023.

XIONG, Ying; SUN, Yulin; XING, Li; HUANG, Ying. Extend cloud to edge with kubeedge. In: IEEE. 2018 IEEE/ACM Symposium on Edge Computing (SEC). [S.l.: s.n.], 2018. P. 373–377.

YANG, Tingting; NING, Jiahong; LAN, Dapeng; ZHANG, Jiawei; YANG, Yang; WANG, Xudong; TAHERKORDI, Amir. Kubeedge wireless for integrated communication and computing services everywhere. **IEEE Wireless Communications**, IEEE, v. 29, n. 2, p. 140–145, 2022.

YOUSEFPOUR, Ashkan; FUNG, Caleb; NGUYEN, Tam; KADIYALA, Krishna; JALALI, Fatemeh; NIAKANLAHIJI, Amirreza; KONG, Jian; JUE, Jason P. All one needs to know about fog computing and related edge computing paradigms: A complete survey. **Journal of Systems Architecture**, v. 98, p. 289–330, 2019. ISSN 1383-7621. DOI: https://doi.org/10.1016/j.sysarc.2019.02.009. Available from: https://www.sciencedirect.com/science/article/pii/S1383762118306349.

ZHAO, Wanlong; HAN, Shuai; HU, Rose; MENG, Weixiao; JIA, Ziqing. Crowdsourcing and Multi-source Fusion Based Fingerprint Sensing in Smartphone Localization. **IEEE Sensors Journal**, PP, p. 1–1, Feb. 2018. DOI: 10.1109/JSEN.2018.2805335.

APPENDIX A – MODIFIED RANDOM WAYPOINT ALGORITHM MOBILITY MODEL FOR IPS

```java
package org.fog.mobilitydata;

import java.io.File;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.io.PrintWriter;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
import java.util.Random;

import org.json.simple.JSONArray;
import org.json.simple.JSONObject;
import org.json.simple.parser.ParseException;

/**
 * @author Victoria Botelho Martins (2023) adapted from Mohammad
    Goudarzi (2021)
 */
public class RandomMobilityGenerator {
    protected Map<Integer, List<Double>> mobilityPositions;
    protected Map<Integer, Double> mobilityPositionsPauseTime;
    protected Map<Integer, Double> mobilityPositionsAngle;
    protected Map<Integer, Double> mobilityPositionsSpeed;
    int NUM_POS = 20;
    double speed;
    double angle;
    double pauseTime;
    boolean directionFlag;
    JSONArray mobilitySpecJSON;

    public RandomMobilityGenerator() {
        mobilityPositions = new HashMap<>();
        mobilityPositionsPauseTime = new HashMap<>();
        mobilityPositionsAngle = new HashMap<>();
        mobilityPositionsSpeed = new HashMap<>();
        mobilitySpecJSON = new JSONArray();
    }

    private static int getRandomNumberInRange(int min, int max) {

        if (min >= max) {
            throw new IllegalArgumentException("max must be greater
                than min");
        }

        Random r = new Random();
        return r.nextInt((max - min) + 1) + min;
    }

    private static boolean positionInRangeCheck(float x, float y) {

        float p1X = -27.60038f;
        float p1Y = -48.51859f;

        float p3X = -27.60045f;
        float p3Y = -48.5185f;

        float p5X = -27.60077f;
        float p5Y = -48.51867f;

        float p6X = -27.60072f;
        float p6Y = -48.51878f;

        final Polygon2D polygon = new Polygon2D();
        polygon.addPoint(p1X, p1Y);
        polygon.addPoint(p3X, p3Y);
        polygon.addPoint(p5X, p5Y);
        polygon.addPoint(p6X, p6Y);

        if (polygon.contains(x, y)) {
            return true;
        } else {
            return false;
        }

    }

    public void createRandomData(int mobilityModel, int user_index,
        String datasetReference, boolean renewDataset)
```

```
            throws IOException, ParseException {
        String fileName = References.dataset_random + user_index + ".csv";
        File tmpDir = new File(fileName);
        boolean exists = tmpDir.exists();
        if (exists && renewDataset) {
            System.out.println("The dataset: " + fileName + " is being overwritten.");
            if (mobilityModel == References.random_walk_mobility_model) {
                MobilityPositionInitiator(References.random_walk_mobility_model, NUM_POS, user_index);
            } else if (mobilityModel == References.random_waypoint_mobility_model) {
                MobilityPositionInitiator(References.random_waypoint_mobility_model, NUM_POS, user_index);
            }
        } else if (!exists) {
            System.out.println("The dataset: " + fileName + " is going to be created for the first time.");
            if (mobilityModel == References.random_walk_mobility_model) {
                MobilityPositionInitiator(References.random_walk_mobility_model, NUM_POS, user_index);
            } else if (mobilityModel == References.random_waypoint_mobility_model) {
                MobilityPositionInitiator(References.random_waypoint_mobility_model, NUM_POS, user_index);
            }
        } else {
            System.out.println("The dataset: " + fileName + " exists already.");
        }

    }

    public void MobilityPositionInitiator(int mobilityModel, int numberOfPositions, int user_index)
            throws IOException, ParseException, org.json.simple.parser.ParseException {
```

```java
        this.mobilityPositions.clear();
        this.mobilityPositionsPauseTime.clear();
        this.mobilityPositionsAngle.clear();
        this.mobilityPositionsSpeed.clear();
        this.mobilitySpecJSON.clear();
        Random r = new Random();

        boolean file = false;

        if (file == false) {
            List<ArrayList<Double>> tempPositions = new ArrayList<
               ArrayList<Double>>();
            tempPositions.add(new ArrayList<Double>());
            Random rd = new Random();
            int low = 10;
            int high = 100;
            int result = rd.nextInt(high - low) + low;

            double positionX;
            double positionY;

            if (References.is_ine_experiment) {
                positionX = References.ine_starting_point_reference
                    [0];
                positionY = References.ine_starting_point_reference
                    [1];
            } else {
                int position = result % 2;
                positionX = References.ens_starting_point_references[
                    position][0];
                positionY = References.ens_starting_point_references[
                    position][1];
            }

            tempPositions.get(0).add(positionX);
            tempPositions.get(0).add(positionY);

            this.angle = getRandomNumberInRange(0, 259);
            directionFlag = true;
            int index = 1;
            int tempIndex = 0;

            this.mobilityPositionsPauseTime.put(0, 0.0);
            this.mobilityPositionsAngle.put(0, angle);
            while (tempIndex < numberOfPositions) {
                int pause_time_multiplier = 3;
                this.mobilityPositionsPauseTime.put(tempIndex, r.nextDouble() * pause_time_multiplier);
                tempIndex++;

            }
            this.mobilityPositions.put(0, tempPositions.get(0));
            JSONObject obj = new JSONObject();
            obj.put("index", 0);
            obj.put("positionX", positionX);
            obj.put("positionY", positionY);

            this.mobilitySpecJSON.add(obj);
            while (index < numberOfPositions) {
                if (this.directionFlag == false || mobilityModel == References.random_walk_mobility_model) {
                    this.angle = getRandomNumberInRange(0, 259);
                    this.directionFlag = true;
                }
                double mobilitySpeed = (double) (getRandomNumberInRange((int) References.MinMobilitySpeed * 100,
                        (int) References.MaxMobilitySpeed * 100)) / 1000;
                tempPositions.add(new ArrayList<Double>());

                double tempPositionX = positionX;
                double tempPositionY = positionY;
                positionX = positionX + (double) (Math.cos(Math.toRadians(this.angle)) * mobilitySpeed) / 10000;
                positionY = positionY + (double) (Math.sin(Math.toRadians(this.angle)) * mobilitySpeed) / 10000;

                if (positionX < -References.environmentLimit) {
                    positionX = -References.environmentLimit;
                    this.directionFlag = false;
                    continue;
                } else if (positionX > References.environmentLimit) {
                    positionX = References.environmentLimit;
```

```
                    this.directionFlag = false;
                    continue;
                }

                if (positionY < -References.environmentLimit) {
                    positionY = -References.environmentLimit;
                    this.directionFlag = false;
                    continue;
                } else if (positionY > References.environmentLimit) {
                    positionY = References.environmentLimit;
                    this.directionFlag = false;
                    continue;
                }

                if (!positionInRangeCheck((float) positionX, (float)
                    positionY)) {
                    System.out.println("positionX: " + positionX + "
                        positionY: " + positionY
                            + " are out of environment bound....going
                                to fix it");
                    positionX = tempPositionX;
                    positionY = tempPositionY;
                }

                tempPositions.get(index).add(positionX);
                tempPositions.get(index).add(positionY);
                this.mobilityPositions.put(index, tempPositions.get(
                    index));
                this.mobilityPositionsAngle.put(index, this.angle);
                this.mobilityPositionsSpeed.put(index, mobilitySpeed)
                    ;
                JSONObject obj1 = new JSONObject();
                obj1.put("index", index);
                obj1.put("positionX", positionX);
                obj1.put("positionY", positionY);
                this.mobilitySpecJSON.add(obj1);

                index++;
            }

            System.out.println("Starting Writing Mobile User
                Information ...");
```

```java

            try (PrintWriter writer = new PrintWriter(new File(References.dataset_random + user_index + ".csv"))) {
                StringBuilder sb = new StringBuilder();
                sb.append("Latitude");
                sb.append(',');
                sb.append("Longitude");
                sb.append('\n');
                writer.write(sb.toString());
                sb.setLength(0);
                for (int i = 0; i < this.mobilityPositions.size(); i++) {
                    sb.append(this.mobilityPositions.get(i).get(0));
                    sb.append(',');
                    sb.append(this.mobilityPositions.get(i).get(1));
                    sb.append('\n');
                    writer.write(sb.toString());
                    sb.setLength(0);
                }

                writer.close();
                System.out.println("done!");

            } catch (FileNotFoundException e) {
                System.out.println(e.getMessage());
            }

            System.out.println("Finished Writing Mobile User Information ...");

        }

    }

}
```

APPENDIX B – CLUSTER FORMATION ALGORITHM

Algorithm 4: Proposed cluster formation algorithm

```
Input  : Array containing fog coordinates
Output : Array containing fog coordinates and regions
  metersPerNode ← AREA/nodes;
  range ← AREA/metersPerNode;
  maxNodesPerCluster ← nodes/metersPerNode;
  clusters ← null;
  {FORMATION}
  for node in nodes do
    if clusters.size() is null then
      adds node to clusters and sets responsible as 0
    else
      if responsible is null then
        tries to get responsible for node and breaks
      end if
      if responsible is still null then
        adds node to clusters
        sets responsible as clusters.size() - 1
        adds node to added array
      end if
    end if
    for node in nodes do
      if nextNode is equal to current then
        continue;
      end if
      if added contains current then
        continue;
      end if
      if the size of clusters[responsible] is less or equal to the maxNodesPerCluster
      then
        if distance between current and nextNode is within range then
          adds nextNode to clusters[responsible]
          adds nextNode to added array
        end if
      else
        if distance between current and nextNode is within range then
          adds node and nextNode to clusters with node as responsible
          adds node and nextNode to added array
        end if
      end if
    end for
  end for
  {OPTIMIZATION}
  for responsible in clusters do
    if the size of clusters[responsible] has more than one node then
      adds group to selected array
    end if
  end for
  return selected
```

ANNEX A – DEFAULT RANDOM WAYPOINT ALGORITHM MOBILITY MODEL FOR OPS FROM THE SIMULATOR

```

package org.fog.mobilitydata;

import org.json.simple.JSONArray;
import org.json.simple.JSONObject;
import org.json.simple.parser.JSONParser;
import org.json.simple.parser.ParseException;

//import  java.awt.geom.
import org.fog.mobilitydata.Polygon2D;

import java.io.*;
import java.util.*;

/**
 * @author Mohammad Goudarzi
 */
public class RandomMobilityGenerator {
    protected Map<Integer, List<Double>> mobilityPositions;
    protected Map<Integer, Double> mobilityPositionsPauseTime;
    protected Map<Integer, Double> mobilityPositionsAngle;
    protected Map<Integer, Double> mobilityPositionsSpeed;
    double speed;
    double angle;
    double pauseTime;
    boolean directionFlag;
    JSONArray mobilitySpecJSON;

    public RandomMobilityGenerator() {
        mobilityPositions = new HashMap<>();  // the list of integer
           contatins the X and Y of one node.
        mobilityPositionsPauseTime = new HashMap<>(); // it shows the
            pause time of mobile user in each geographical point
        mobilityPositionsAngle = new HashMap<>(); //it shows the
           direction of the move for the next period of the time
        mobilityPositionsSpeed = new HashMap<>(); //it shows the
           speed of the move for the next period of the time
        mobilitySpecJSON = new JSONArray();
    }

    private static int getRandomNumberInRange(int min, int max) {

        if (min >= max) {
            throw new IllegalArgumentException("max must be greater
                than min");
        }

        Random r = new Random();
        return r.nextInt((max - min) + 1) + min;
    }

    private static boolean positionInRangeCheck(float x, float y) {
        float topLeftXEnv = -37.813046f;
        float topLeftYEnv = 144.951380f;

        float downLeftXEnv = -37.821229f;
        float downLeftYEnv = 144.955039f;


        float topRightXEnv = -37.807397f;
        float topRightYEnv = 144.971062f;

        float downRightXEnv = -37.815136f;
        float downRightYEnv = 144.975044f;

        final Polygon2D polygon = new Polygon2D();
        polygon.addPoint(topLeftXEnv, topLeftYEnv);
        polygon.addPoint(downLeftXEnv, downLeftYEnv);
        polygon.addPoint(topRightXEnv, topRightYEnv);
        polygon.addPoint(downRightXEnv, downRightYEnv);

        if (polygon.contains(x, y)) {
            return true;
        } else {
            return false;
        }


    }

    public void createRandomData(int mobilityModel, int user_index,
       String datasetReference, boolean renewDataset) throws
       IOException, ParseException {
        // To check different mobility models, if you applied other
           mobility models, they can be customized here
        String fileName = References.dataset_random + user_index + ".
           csv";
        File tmpDir = new File(fileName);
        boolean exists = tmpDir.exists();
        if (exists && renewDataset) {
            System.out.println("The dataset: " + fileName + " is
               being overwritten.");
            if (mobilityModel == References.
               random_walk_mobility_model) {
                MobilityPositionInitiator(References.
                   random_walk_mobility_model, 100, user_index);
            } else if (mobilityModel == References.
               random_waypoint_mobility_model) {
                MobilityPositionInitiator(References.
                   random_waypoint_mobility_model, 100, user_index);
            }
        } else if (!exists) {
            System.out.println("The dataset: " + fileName + " is
               going to be created for the first time.");
            if (mobilityModel == References.
               random_walk_mobility_model) {
                MobilityPositionInitiator(References.
                   random_walk_mobility_model, 100, user_index);
            } else if (mobilityModel == References.
               random_waypoint_mobility_model) {
                MobilityPositionInitiator(References.
                   random_waypoint_mobility_model, 100, user_index);
            }
        } else {
            System.out.println("The dataset: " + fileName + " exists
               already.");
            // DO NOTHING
        }


    }

```

```java
    public void MobilityPositionInitiator(int mobilityModel, int
        numberOfPositions, int user_index) throws IOException,
            ParseException, org.json.simple.parser.ParseException {
        this.mobilityPositions.clear();
        this.mobilityPositionsPauseTime.clear();
        this.mobilityPositionsAngle.clear();
        this.mobilityPositionsSpeed.clear();
        this.mobilitySpecJSON.clear();
        Random r = new Random();

        boolean file = false;


        if (file == false) {
            List<ArrayList<Double>> tempPositions = new ArrayList<
                ArrayList<Double>>();
            tempPositions.add(new ArrayList<Double>());
            double positionX = References.lat_reference;
            double positionY = References.long_reference;
            tempPositions.get(0).add(positionX);
            tempPositions.get(0).add(positionY);

            this.angle = getRandomNumberInRange(0, 259);
            directionFlag = true;
            int index = 1;
            int tempIndex = 0;

            this.mobilityPositionsPauseTime.put(0, 0.0);
            this.mobilityPositionsAngle.put(0, angle);
            while (tempIndex < numberOfPositions) {
                int pause_time_multiplier = 3;
                this.mobilityPositionsPauseTime.put(tempIndex, r.
                    nextDouble() * pause_time_multiplier);
                tempIndex++;

            }
            this.mobilityPositions.put(0, tempPositions.get(0));
            JSONObject obj = new JSONObject();
            obj.put("index", 0);
            obj.put("positionX", positionX);
            obj.put("positionY", positionY);

            this.mobilitySpecJSON.add(obj);
            while (index < numberOfPositions) {
                if (this.directionFlag == false || mobilityModel ==
                   References.random_walk_mobility_model) {
                    this.angle = getRandomNumberInRange(0, 259); //
                       Random direction.
                    this.directionFlag = true;
                }
                double mobilitySpeed = (double) (
                   getRandomNumberInRange((int) References.
                   MinMobilitySpeed * 100,
                        (int) References.MaxMobilitySpeed * 100)) /
                          100; // meter/seconds
                tempPositions.add(new ArrayList<Double>());

                //positionX = positionX + (double) (Math.cos(Math.
                   toRadians(angle)) * speed) * (time -
                   mobilityPositionsPauseTime.get(index - 1));
                //positionY = positionY + (double) (Math.sin(Math.
                   toRadians(angle)) * speed) * (time -
                   mobilityPositionsPauseTime.get(index - 1));
                double tempPositionX = positionX;
                double tempPositionY = positionY;
                positionX = positionX + (double) (Math.cos(Math.
                   toRadians(this.angle)) * mobilitySpeed) / 1000; //
                    divided by 1000 to change the values to KM
                positionY = positionY + (double) (Math.sin(Math.
                   toRadians(this.angle)) * mobilitySpeed) / 1000; //
                    divided by 1000 to change the values to KM

                if (positionX < -References.environmentLimit) {
                    positionX = -References.environmentLimit;
                    this.directionFlag = false;
                    continue;
                } else if (positionX > References.environmentLimit) {
                    positionX = References.environmentLimit;
                    this.directionFlag = false;
                    continue;
                }

                if (positionY < -References.environmentLimit) {
                    positionY = -References.environmentLimit;
                    this.directionFlag = false;
                    continue;
                } else if (positionY > References.environmentLimit) {
                    positionY = References.environmentLimit;
                    this.directionFlag = false;
                    continue;
                }

                if (!positionInRangeCheck((float) positionX, (float) positionY)) {
                    System.out.println("positionX: " + positionX + " positionY: " + positionX + " are out of environment bound....going to fix it");
                    positionX = tempPositionX;
                    positionY = tempPositionY;
                }


                tempPositions.get(index).add(positionX);
                tempPositions.get(index).add(positionY);
                this.mobilityPositions.put(index, tempPositions.get(index));
                this.mobilityPositionsAngle.put(index, this.angle);
                this.mobilityPositionsSpeed.put(index, mobilitySpeed);
                JSONObject obj1 = new JSONObject();
                obj1.put("index", index);
                obj1.put("positionX", positionX);
                obj1.put("positionY", positionY);
                this.mobilitySpecJSON.add(obj1);

                index++;
            }

            // File input path
            System.out.println("Starting Writing Mobile User Information ...");

            try (PrintWriter writer = new PrintWriter(new File(References.dataset_random + user_index + ".csv"))) {
                StringBuilder sb = new StringBuilder();
                sb.append("Latitude");
```

```
                    sb.append(',');
                    sb.append("Longitude");
                    sb.append('\n');
                    writer.write(sb.toString());
                    sb.setLength(0); // clear stringbuilder
                    for (int i = 0; i < this.mobilityPositions.size(); i
                       ++) {
                        sb.append(this.mobilityPositions.get(i).get(0));
                        sb.append(',');
                        sb.append(this.mobilityPositions.get(i).get(1));
                        sb.append('\n');
                        writer.write(sb.toString());
                        sb.setLength(0); // clear stringbuilder
                    }

                    writer.close();
                    System.out.println("done!");

                } catch (FileNotFoundException e) {
                    System.out.println(e.getMessage());
                }

                System.out.println("Finished Writing Mobile User
                    Information ...");

            }


        }

}
```