UNIVERSIDADE DE PASSO FUNDO
Programa de Pós-Graduação em Computação Aplicada

Dissertação de Mestrado

MONITORAMENTO DE ENERGIA ELÉTRICA EM RESIDÊNCIAS NO AMBIENTE DE CIDADES INTELIGENTES

GIOVANI ANDRÉ RIZZARDI

UPF

UNIVERSIDADE DE PASSO FUNDO
INSTITUTO DE CIÊNCIAS EXATAS E GEOCIÊNCIAS
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO APLICADA

MONITORAMENTO DE ENERGIA ELÉTRICA EM RESIDÊNCIAS NO AMBIENTE DE CIDADES INTELIGENTES

Giovani André Rizzardi

Dissertação apresentada como requisito parcial à obtenção do grau de Mestre em Computação Aplicada na Universidade de Passo Fundo.

Orientador: Prof. Dr. Marcelo Trindade Rebonatto
Coorientador: Prof. Dr. Luís Eduardo Schardong Spalding

Passo Fundo
2020

CIP – Catalogação na Publicação

R627m Rizzardi, Giovani André
Monitoramento de energia elétrica em residências no ambiente de cidades inteligentes / Giovani André Rizzardi. – 2020.
82 f. : il. ; 30 cm.

Orientador: Prof. Dr. Marcelo Trindade Rebonatto.
Coorientador: Prof. Dr. Luís Eduardo Schardong Spalding.
Dissertação (Mestre em Computação Aplicada) – Universidade de Passo Fundo, 2020.

1. Automação residencial. 2. Cidades Inteligentes. 3. Software de aplicação. 4. Redes elétricas inteligentes. I. Rebonatto, Marcelo Trindade, orientador. II. Spalding, Luís Eduardo Schardong, coorientador. III. Título.

CDU: 004.4

Catalogação: Bibliotecário Luís Diego Dias de S. da Silva – CRB 10/2241

UPF | PPGCA
Programa de Pós-Graduação
em Computação Aplicada
Instituto de Ciências Exatas e Geociências | ICEG

ATA DE DEFESA DO TRABALHO DE CONCLUSÃO DE CURSO DO ACADÊMICO

GIOVANI ANDRÉ RIZZARDI

Aos vinte e sete dias do mês de março do ano de dois mil e vinte, às quatorze horas, realizou-se, no prédio D01 sala 01, da Universidade de Passo Fundo (UPF), a sessão pública de defesa do Trabalho de Conclusão de Curso "Monitoramento de energia elétrica em residências no ambiente de cidades inteligentes", de autoria do **Giovani André Rizzardi**, acadêmico do Curso de Mestrado em Computação Aplicada do Programa de Pós-Graduação em Computação Aplicada – PPGCA. Segundo as informações prestadas pelo Conselho de Pós-Graduação e constantes nos arquivos da Secretaria do PPGCA, o aluno preencheu os requisitos necessários para submeter seu trabalho à avaliação. A banca examinadora foi composta pelos doutores Marcelo Trindade Rebonatto, Luiz Eduardo Schardong Spalding, Roberto dos Santos Rabello e Higor Amario de Souza. Concluídos os trabalhos de apresentação e arguição, a banca examinadora considerou o candidato APROVADO. Foi concedido o prazo de até quarenta e cinco (45) dias, conforme Regimento do PPGCA, para o acadêmico apresentar ao Conselho de Pós-Graduação o trabalho em sua redação definitiva, a fim de que sejam feitos os encaminhamentos necessários à emissão do Diploma de Mestre em Computação Aplicada. Para constar, foi lavrada a presente ata, que vai assinada pelos membros da banca examinadora e pela Coordenação do PPGCA.

Prof. Dr. Marcelo Trindade Rebonatto– UPF
Presidente da Banca Examinadora
(Orientador)

Prof. Dr. Luiz Eduardo Schardong Spalding
(Coorientador)

Prof. Dr. Roberto dos Santos Rabello – UPF
(Avaliador Interno)

Prof. Dr. Higor Amario de Souza -USP
(Avaliador Externo)

Prof. Dr. Rafael Rieder
Coordenador do PPGCA

AGRADECIMENTOS

Agradecer primeiramente à minha família por sempre me apoiar em todos os momentos. Agradeço à UPF pela oportunidade que me foi dada através da bolsa auxílio. Um agradecimento especial aos meus orientadores Dr. Marcelo Trindade Rebonatto e Dr. Luís Eduardo Schardong Spalding que foram sempre importantíssimos para o condução deste trabalho. Agradeço também ao Me. Matheus Janio Mella, Rogério Zimmermann e MUX energia, por disponibilizarem o protótipo de monitoramento de energia elétrica. Por fim, mas não menos importante, um agradecimento ao engenheiro elétrico da UPF, Rangel Casanova Daneli, por apresentar as informações de energia da UPF, contribuindo com várias ideias para este trabalho.

MONITORAMENTO DE ENERGIA ELÉTRICA EM RESIDÊNCIAS NO AMBIENTE DE CIDADES INTELIGENTES

RESUMO

O conceito de Cidades Inteligentes é um termo que se espalha muito rápido pela sociedade. Cada vez mais sistemas tecnológicos capazes de agilizar processos são empregados em cidades para trazer segurança e qualidade de vida para os cidadãos. Um exemplo disso são os sistemas integrados de monitoramento urbano, como por exemplo tráfego e meio ambiente, geralmente instalados em plataformas que oferecem suporte aos mais variados tipos de aplicações para um ambiente de cidades inteligentes. Em residências, o comportamento não é diferente: sistemas de monitoramento cada vez mais automatizados são instalados para trazer autonomia e segurança para o cidadão. Monitoramento de energia elétrica em residências é área que necessita de uma atenção maior em cidades inteligentes, esclarecendo cada vez mais para as pessoas a importância sobre o consumo consciente de energia. O presente trabalho tem como objetivo incluir, em uma plataforma de suporte a aplicações de cidades inteligentes, um serviço que integre um protótipo de monitoramento de energia elétrica. Os dados gerados de uma residência são manipulados e disponibilizados através de *dashboards*, baseando-se em indicadores de sustentabilidade propostos pela ISO 37120. Validou-se que as informações geradas pelos indicadores da ISO podem ajudar o usuário a ter consciência sobre o consumo elétrico da residência. Espera-se que os dados apresentados neste trabalho ajudem o consumidor final a ter uma visão mais detalhada da sua rede elétrica, além de auxiliar em um consumo consciente de energia.

Palavras-Chave: Casa Inteligente, Consumo Consciente de Energia Elétrica, Monitoramento Elétrico em Residências, InterSCity.

ELECTRICAL MONITORING AT HOMES IN A SMART CITIES ENVIRONMENT

ABSTRACT

The concept of Smart Cities is a term that spreads very fast through society, a lot of technological systems capable of streamlining processes are employed in cities to bring security and quality of life to citizens. An example are integrated urban monitoring systems, such as traffic and the environment, generally installed on platforms that support the most varied types of applications needed for a smart city environment. In homes, the behavior is no different, a lot of automated monitoring systems are installed to bring more autonomy and security to the citizen. Monitoring electricity in homes is something that needs more attention in smart cities, clarifying more and more for people the importance of conscious energy consumption. The present work aims to include, in a support platform for smart city applications, a service that integrates an electric energy monitoring prototype. The data generated from a residence will be manipulated and made available in dashboards, based on sustainability indicators proposed by ISO 37120. The electricity information generated by the ISO indicators can help the user to be aware of the electrical consumption of the residence. It is hoped that the data presented in this work will help the final consumer to have a more detailed view of their electrical network, in addition to help in a conscious consumption of energy.

Keywords: Smart Home, Conscious Consumption of Electricity, Electrical Monitoring in Homes.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

MSL – Mapeamento Sistemático da Literatura
IOT – Internet of Things
SDK – Software Development Kit
REST – Representational State Transfer
SO – Sistema operacional
SOA – Service-Oriented Architecture
HTTP – Hypertext Transfer Protocol
RFID – Radio-Frequency IDentification
NFC – Near Field Communication
MQTT – Message Queue Telemetry Transport
GIS – Geographic Information System
BIM – Building Information Modeling
SIM – System Information model
API – Application Program Interfaces
GE – Generic Enablers
NGSI – Next Generation Service Interfaces
PDP – Policy Decision Point
PEP – Policy Enforcement Point
API – Application Program Interface
UUID – Identificador Universal Único
KWH – Quilowatt-hora
V – Volt
A – Amperes

SUMÁRIO

1. INTRODUÇÃO

Cidades Inteligentes agregam projetos que visam melhorar o espaço urbano, com o uso de Tecnologia de Informação e Comunicação(TICs) e Internet das Coisas(IoT) [1]. Novos projetos e produtos podem ser criados para agregar valor em áreas técnicas de comunicação entre dispositivos, Big Data e gestão governamental (ajuda na tomada de decisões com base nos dados gerados). O principal objetivo é criar um ambiente sustentável, com melhores condições de vida, além de ajudar na criação de uma economia criativa e sustentável [1].

De acordo com a *United Nations Population Fund*, em 2030, estima-se que 66% da população (5 bilhões) estejam vivendo em áreas urbanas [2]. Estes números ampliam a quantidade de projetos para cidades inteligentes, com o objetivo de criar soluções que utilizam TICs para melhorar operações dentro de uma cidade de forma inteligente, otimizada, com possibilidade de tomada de decisões em tempo real [2]. O papel de uma cidade inteligente é colaborar com a resolução de desafios e problemas urbanos, propondo novos desafios e tecnologias que consigam entregar uma melhor qualidade de vida para os cidadãos. Todo esse esforço é feito para gerar uma infraestrutura a favor do meio ambiente e que seja otimizada para evitar problemas do cotidiano, como engarrafamentos e alagamentos urbanos causadas, por exemplo, pelo descarte incorreto de lixo [3].

Com o aumento do uso de TICs, principalmente após a criação de redes *wireless* com baixo consumo energético, grandes quantidades de dados começaram a ser geradas. Devido a isso, arquiteturas para tratamento destes dados começaram a surgir de modo que seja possível facilitar a comunicação entre essas redes e outras tecnologias. Com estas arquiteturas, a criação de aplicações para cidades inteligentes fica mais ágil, não sendo necessário desenvolver toda a infraestrutura do início, utilizando funcionalidades já existentes. Desta forma, é possível desenvolver diversos sistemas que podem beneficiar não somente o cidadão, mas empresas e também governos, a tomarem decisões com base no processamento desses dados, melhorando a qualidade de vida dentro de uma cidade [4].

Com o conceito de Cidades inteligentes se expandindo, grandes parcerias se formam para discutir soluções tecnológicas para os reais problemas de uma cidade. É o caso da *Smart City Council*, formado por um conjunto de grandes *players* da tecnologia, focados em desenvolver padrões e modelos a serem empregados em uma cidade inteligente. Outras grandes companhias apostam no desenvolvimento dessas soluções, como as empresas IBM, Siemens, Cisco, entre outras [5].

O conceito de cidades inteligentes serviu de base para a criação de vários outros temas, como *Smart Buildings* focados em desenvolver técnicas sustentáveis para constru-

ção de imóveis, visando o controle de várias áreas da residência via sensores e monitoradores [6]. *Smart Farms* são técnicas sustentáveis com o uso da tecnologia para auxiliar o produtor agrícola na fazenda [7].

Um *Smart building* prevê a criação de um ambiente confortável e produtivo, com os usuários usufruindo de sistemas de monitoramento e controle das partes do imóvel, tais como controle de energia elétrica, ventilação, segurança (meios humanos, tais como furto e roubo, e meios naturais como fogo). Além disso, uma das metas dos *Smart buildings* é a redução da emissão de carbono na atmosfera, gerando um ecossistema mais sustentável e limpo [6].

Smart Grids são conhecidos por criar sistemas de redes elétricas de forma sustentável, econômica e confiável, através de tecnologias com pouca agressão ao meio ambiente [8], porém isso requer que toda a rede de energia existente seja repensada, principalmente nos centros de distribuição, onde são aplicadas as maiores mudanças. Estas novas estruturas devem ter sistemas de TICs para que seja possível realizar mudanças de acordo com o surgimento de necessidades [9].

As cidades inteligentes estão cada vez mais integrando sensores, dados e usuários nos mais variados tipos de aplicações existentes, tais como transporte público, monitoramento climático, saúde, entre outros. Algumas aplicações possuem plataformas sustentando a infraestrutura do aplicativo [10, 11].

Os sistemas de monitoramento atuais em ambiente de cidades inteligentes como trânsito ou saúde por exemplo, tem como um dos principais objetivos, a detecção de eventuais problemas nessas áreas, ajudando a conscientizar os cidadãos sobre as consequências destes problemas [12]. Informações a respeito do consumo de energia de uma determinada região de uma cidade, ou simplesmente de uma única residência, beneficiaria a área de consumo consciente de energia [13].

Plataformas multipropósitos para cidades inteligentes costumam suportar serviços que trabalham com diversos tipos de dados provenientes de sensores com diferentes finalidades e variedades. Nas residências, por exemplo, existem monitoramentos na área da saúde, ajudando a detectar pontos relacionados a sinais vitais dos moradores e na automação, usando sensores e atuadores conectados à aplicativos em *smartphones* para colaborar com uma casa inteligente [14, 15].

As plataformas de cidades inteligentes muitas vezes são desenvolvidas para possibilitar a tomada de decisões e resolver determinados problemas dentro de uma cidade. Em uma residência, o comportamento não é diferente, sendo necessária uma boa arquitetura para acomodar todos os componentes de uma construção inteligente. Porém, em alguns casos existem dificuldades para introduzir o conceito de *Smart Buildings* quando a casa em questão é antiga, sendo necessário um grande planejamento para torná-la inteligente, podendo gerar um custo excessivamente alto.

No Brasil, desconhece-se uma plataforma de software para facilitar a criação de cidades inteligentes que integre um serviço para monitoramento de energia elétrica residencial. Sendo assim, esta área sofre um déficit de informações a respeito de energia dentro de um ambiente de cidades inteligentes.

A implementação de um serviço receptor de dados de energia elétrica residencial, ligado a uma plataforma de cidades inteligentes, poderia impulsionar a linha de pesquisa em energia elétrica residencial. Este estudo pode ser ampliado posteriormente para outros ambientes da cidade, abrindo caminhos para novos aplicativos de monitoramento de energia dentro de uma cidade inteligente.

O deficit de informações sobre energia elétrica poderia ser menor, se as companhias de energia, fornecessem um *feedback* do consumo elétrico diário para os seus clientes[16]. O objetivo deste trabalho visa minimizar estes problemas, utilizando um equipamento medidor de consumo de energia elétrica em residências para gerar dados e disponibilizar em uma plataforma de cidades inteligentes. A partir dos dados obtidos, é apresentado uma aplicação que reúne informações acerca do consumo elétrico residencial, seguindo os indicadores de sustentabilidade previstos na ISO 37120, obtendo um ambiente de cidades inteligentes com monitoramento da energia elétrica em residências.

A estrutura deste trabalhado está definida da seguinte forma. No capítulo 2 é apresentada a revisão bibliográfica e um mapeamento sistemático desenvolvido, seguido pelo capítulo 3 que apresenta os materiais e métodos utilizados neste trabalho. A implementação, descrita no capítulo 4 especifica os recursos utilizados juntamente com as funcionalidades implementadas. O capítulo 5 quantifica os resultados gerados neste trabalho, seguido pelas considerações finais no capítulo 6.

2. REVISÃO BIBLIOGRÁFICA

Nesta seção, é apresentada a bibliografia usada que serviu como base científica para o trabalho, tais como os detalhes do mapeamento sistemático da literatura, usado para buscar plataformas de cidades inteligentes e a utilização de indicadores de qualidade e sustentabilidade.

2.1 MAPEAMENTO SISTEMÁTICO

Com o objetivo de compreender as tecnologias empregadas em cidades inteligentes, foi feito um Mapeamento Sistemático da Literatura (MSL) visando encontrar plataformas e arquiteturas que oferecem, de alguma forma, suporte a aplicações voltadas para ambientes de cidades inteligentes. No estudo foram comparados conceitos, objetivos e arquitetura das plataformas e outros aspectos.

Um MSL visa dar suporte às diretrizes da pesquisa, concentrando-se na busca, refinamento e seleção de outras pesquisas que ajudem a sanar os objetivos [17]. Este MSL visa encontrar e compreender as plataformas existentes que oferecem suporte a novas aplicações em cidades inteligentes. Sendo assim, é necessário obter um conhecimento mais aprofundado das arquiteturas, a fim de ter uma base de conhecimento para escolher qual a mais vantajosa para ser usada no projeto. As questões que motivaram uma mapeamento sistemático foram:

- Q1. Qual a plataforma que atende melhor aos requisitos necessários deste projeto de pesquisa?

- Q2. Esta plataforma vai proporcionar suporte ao usuário em caso de necessidade?

O estudo de Tomas et. al. propõe uma revisão sistemática a partir de arquiteturas de cidades inteligentes, empregando as plataformas existentes até o ano de 2012. Na revisão feita, foram especificadas 11 plataformas, em seguida analisadas, para gerar um comparativo entre elas, analisando alguns pontos, tais como capacidade de monitoramento em tempo real, interoperabilidade de objetos, políticas sustentáveis, aspectos sociais, ubiquidade, segurança e outros aspectos [18].

O estudo de Santana et. al. procura evidenciar o conceito de Cidades inteligentes, analisando plataformas que oferecem suporte para aplicações IoT, além de descrever as tendências de pesquisa na área de cidades inteligentes. Nessa análise, são identificados requisitos funcionais e não funcionais a respeito de cada plataforma [19].

A *string* de busca usada para este trabalho foi criada com base na MSL de Tomas et. al., onde possui objetivos semelhantes de encontrar arquiteturas, plataformas, *middlewares* para cidades inteligentes [18]. A diferença desta pesquisa é a procura por plataformas estruturadas que ofereçam suporte a novas aplicações de cidades inteligentes dos mais variados objetivos, sem a necessidade de contribuições científicas sobre os dados pesquisados. Os critério de inclusão e exclusão de artigos baseiam-se em exclusão por intervalo fora de 2012-2019, por título, resumo e relevância.

A *string* de busca tem por objetivo, encontrar nas bibliotecas digitais, os artigos que se encaixam nas condições impostas. A *string* de busca possui o seguinte formato: *(smart city OR smart cities OR smart digital OR smartcities OR urban environment OR smart features) AND (internet of things OR iot OR heterogeneous sensors OR sensors) AND (architecture OR middleware OR platform)*.

A busca foi feita em quatro bibliotecas digitais, tais como: IEEExplore, Springer, Science Direct e ACM. Em cada uma das bibliotecas, a *string* teve de ser adaptada para o formato da plataforma, de forma que faça uma busca semelhante as outras bibliotecas. Na Tabela 1 estão descritos os passos que foram seguidos para a obtenção dos estudos e a sua quantidade.

Tabela 1. Etapas do processo de Refinamento.

Passo	Descrição	Artigos Excluídos	Número de Artigos
1	Busca completa nas bases	0	6817
2	Exclusão de estudos fora do intervalo entre 2012-2019	3929	2888
3	Exclusão de estudos por títulos	2798	90
4	Exclusão de estudos por resumo/abstract	62	28
5	Exclusão de estudos por relevância	15	13

Após a busca dos artigos no primeiro passo, excluiu-se os artigos que foram publicados fora do intervalo entre o ano 2012 e 2019. Pois na revisão sistemática de Tomas et. al. a busca atingiu resultados até o ano de 2012 [18].

Com o intervalo de datas já refinado, foi feita a leitura dos títulos dos artigos, sendo que aqueles que não condiziam com o objetivo da pesquisa, eram excluídos da pesquisa. O mesmo ocorreu com os critérios de relevância, onde era feito a leitura de todo o artigo, buscando entender se a plataforma realmente está sendo utilizada. Esta etapa contribuiu para reduzir a quantidade de artigos a serem analisados, sendo excluídos somente pela leitura do título.

Com a busca já refinada, foram registradas oito plataformas que atenderam aos requisitos da busca. A Tabela 2 contém os nomes das plataformas encontradas. Durante a execução da pesquisa e refinamento, foram identificados vários artigos em que descreviam

plataformas desenhadas para suprir objetivos específicos, servindo apenas para atender um único propósito de pesquisa e por este motivo, foram excluídas da busca.

Tabela 2. Plataformas encontradas.

ID	NOME	REFERÊNCIAS PRINCIPAIS
1	InterSCity	[20, 21]
2	Dimmer	[22, 23]
3	Almanac	[24, 25]
4	SmartSantander	[26, 27]
5	CiVitas	[28]
6	Gambas	[29]
7	Kaa IoT	[30]
8	Fiware	[31, 32]

Durante a execução do MSL, quando uma plataforma era encontrada, dedicava-se um tempo para pesquisa em outras fontes, averiguando se aquela plataforma possuía os requisitos necessários para estar nesta pesquisa. Deste modo, quando era encontrado um novo artigo referenciando a plataforma já conhecida, este artigo era automaticamente movido para a lista dos artigos aceitos na pesquisa.

A seção 2.2 explica cada uma das plataformas encontradas na MSL, descrevendo brevemente sua arquitetura. Durante a análise das plataformas, notou-se a necessidade de fazer um detalhamento mais aprofundado de algumas delas.

2.2 PLATAFORMAS ANALISADAS

As plataformas de suporte a aplicações de cidades inteligentes são desenvolvidas para que seja possível captar dados dentro de um perímetro através de vários tipos de sensores e atuadores. Os dados gerados são transportados para centros de processamento e mineração de dados, que conseguem extrair informações [33].

A coleta de dados acontece pelos mais variados tipos de sensores. Cada plataforma possui um método para integrar os sensores em seus sistemas. Estes sensores geralmente ficam conectados em lugares que possuam acesso a energia e rede, para que seja possível transmitir os dados. Dispositivos móveis também podem ser utilizados, uma vez que possuem vários sensores integrados [27].

Cada plataforma diverge das outras no modo de processar os dados, mas possuem em comum, um centro de processamento, geralmente instalado em servidores para que seja processado os dados que chegam dos sensores, possibilitando a geração de informações para o usuário [34].

2.2.1 SmartSantander

Sanchez descreve o SmartSantander como uma plataforma para suportar aplicações em larga escala voltadas para cidades inteligentes. Inicialmente testada na cidade de Santander, no norte da Espanha, sua rede já é integrada por mais de 20000 objetos e sensores IoT [27] .

A SmartSantader é capaz de se comunicar com diferentes serviços e dispositivos, usando diferentes tecnologias. A plataforma possui um serviço de gerenciamento para *testbed*(plataforma para a realização de testes de teorias científicas e ferramentas computacionais) que busca facilitar a gerência de alguns pontos, tais como configuração dinâmica, configurações *plug-and-play* e gerenciamento de falhas no *framework* [26].

Alguns casos de uso foram descritos para a plataforma, tais como monitoramento de ambiente, gerenciamento de estacionamento e orientação ao motorista, irrigação precisa de parques e jardins e realidade aumentada. São tarefas possíveis graças a precisão dos dados dos sensores espalhados pela cidade [27].

2.2.2 Civitas

Civitas é um *middleware* que se comporta como um ecossistema com padrões que facilitam o suporte de novas aplicações e projetos na área de mobilidade urbana. Os dados são coletados de sensores externos e através de algoritmos eficientes, eles são transformados em informações úteis para um aplicativo de cidade inteligente. O *middleware* consegue interagir com diversos tipos de dados, para assim criar um ecossistema que coleta e processa dados [28].

A infraestrutura é geralmente suportada por uma entidade pública ou governamental. Os dados são capturados por um aplicativo chamado Civitas *plug*, licenciado e compatível com smartfones e outras plataformas.

O *middleware* foi projetado para respeitar certos princípios, onde tudo dentro da plataforma é transformado em objetos para que sua utilização seja mais fácil, focando na reutilização do *software*, além de ter suporte para dados do tipo áudio e vídeo. Os usuários conectam-se através do dispositivo Civitas *plug*, que garante privacidade e segurança na conexão entre o dispositivo do usuário e o *hardware* [28].

2.2.3 Gambas

Gambas é um *middleware* aberto para aplicações de IoT e cidades inteligentes, envolvendo padrões de comunicação e componentes orientados para manter uma estrutura com privacidade e eficiência. A plataforma consegue integrar sistemas controlados por

outros *middlewares*, fornecendo maneiras simplificadas de lidar com o compartilhamento de informações na plataforma [29].

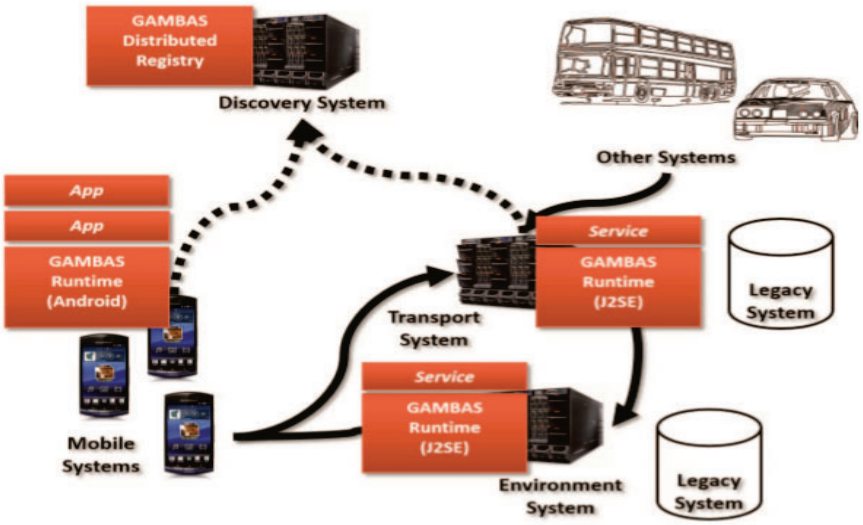

Figura 1. Arquitetura do Gambas [29].

A Figura 1 demonstra a visão geral da arquitetura do Gambas, sendo constituído por três sistemas de coleta e comunicação: *Mobile system*, representado por celulares com sistema operacional Android, onde usa os sensores presentes nos dispositivos para captar dados, enviando para um servidor. *J2SE Runtime* tem o objetivo de capturar os dados dos celulares e modificá-los para o formato que é necessário nos servidores. *Discovery system* permite que os dispositivos celulares e os servidores abram um canal de comunicação usando BASE [35] e publiquem metadados [29].

2.2.4 Kaa IoT Plataform

Kaa Project é um *middleware* de código aberto, multi-propósito para suportar aplicações de cidades inteligentes e IoT. Segundo a Empresa *Kaa IoT Technologies*, o *middleware* possui *toolkits* que ajudam o desenvolvimento do projeto a minimizar custos e riscos [36].

A plataforma possui um sistema de gerenciamento que minimiza a codificação dos objetos conectados para *plug-and-play*. O *middleware* possui SDKs que são compatíveis com vários tipos de dispositivos. Também é responsável por manter a conexão e troca de mensagens entre o dispositivo e o servidor e por isso é integrado ao dispositivo. A aplicação do Kaa trata os sensores de acordo com o tipo de dado que eles entregam. Por exemplo, se existir dois sensores diferentes para medição de pressão e ambos entregarem o mesmo tipo de dado, então a aplicação tratará os dois sensores como sendo iguais [36].

Os servidores do Kaa IoT já fornecem as funcionalidades *back-end* iniciais para a aplicação e podem ser configurados para suportarem aplicações de grande escala e apli-

cações críticas. Também possuem interfaces para integrações de sistemas para gerenciamento de dados. Desta forma, o projeto já entrega uma estrutura inicial para o desenvolvimento.

O *middleware* possui suporte para os sistemas operacionais(SO) Android, Ios, Windows, Linux e Raspberry Pi. Porém, segundo a Kaa, instalar em outro SO não é difícil [36]. O SDK foi projetado para as linguagens *Java*, *C*, *C++* e *Objective-C*. A escolha de qual usar dependerá do suporte do SO que o desenvolvedor for usar. Kaa possui suporte para vários tipos de comunicação, sendo que o desenvolvedor deverá escolher algum tipo de protocolo de transporte de dados que o *middleware* oferece. O Kaa *server* utiliza o protocolo *Representational State Transfer* (REST) para comunicação com outros servidores [30].

2.2.5 Fiware

Segundo Ferreira, Fiware é uma plataforma de código aberto que oferece suporte para aplicações de IoT. Possui uma grande quantidade de APIs para que possa conectar, coletar e analisar os dados de uma aplicação que são gerados a partir de sensores [31].

A plataforma possui uma gama de interfaces abertas que ajudam o desenvolvedor no seu projeto, além de ser modular. Essa plataforma é usada em diversos casos de cidades inteligentes na Europa. No Brasil por exemplo, a cidade de Natal, no estado do Rio Grande do Norte utiliza em um projeto, alguns componentes do Fiware, justamente por ser de código aberto e fácil modularidade, podendo ser acrescentado sem complexidade [11].

A plataforma Fiware conta também com outro projeto em paralelo. Fiware Lab é um *testbed* onde é possível implantar projetos da plataforma para testes mais rígidos. Por ser bastante modular, a plataforma se ajusta facilmente para estratégias de desenvolvimento ágil, acelerando a implementação de novos sistemas na plataforma [37].

2.2.6 InterSCity

Del Esposte et. al. descreve InterSCity como uma plataforma para desenvolvimento de aplicações, pesquisas e tecnologias que são empregadas na infraestrutura de cidades inteligentes. Um dos focos da plataforma é o desenvolvimento visando reutilização de código, uma vez que o projeto possui código aberto e foi desenvolvida incrementalmente utilizando metodologias ágeis [20, 21].

Utilizando arquitetura em microsserviços, a plataforma pode suportar o desenvolvimento de sistemas que integram a maior parte das áreas de uma cidades inteligentes, tais como transporte urbano, segurança pública e monitoramento do tempo. A plataforma pos-

sui um alto nível de serviços na nuvem para gerenciamento de sensores IoT padronizados, além de controle de processamento de dados.

A plataforma InterScity foi projetada para suportar aplicações com grande quantidade de usuários, dados e serviços. Um exemplo de aplicação é o *Smart Parking*, que foi desenvolvido na plataforma com o objetivo de ajudar os motoristas a acharem vagas disponíveis em estacionamentos públicos próximos. As informações são baseadas em dados de simulação de sensores que estão instalados nos estacionamentos que detectam a presença de veículos nas vagas [20].

2.2.7 Dimmer

O projeto *District Information Modeling and Management for Energy Reduction* (DIMMER) visa criar uma plataforma para visualização e simulação do uso de energia a nível urbano, coletando, processando e projetando estas informações. A plataforma também coleta informações de monitoramento em tempo real, enviadas por sensores instalados em edifícios e nas redes de distribuição, podendo prever possíveis problemas como aquecimento em redes de energia [22].

O projeto possui três tipos de usuários finais: Administradores públicos, onde visa buscar redução dos custos com consumo de energia; Gerentes de construções, onde busca melhorar a operabilidade dos edifícios de acordo com o uso pretendido; Profissionais de energia, onde procuram melhorar a distribuição de energia com maior eficiência [38].

Os projetos foram testados inicialmente em Manchester, no Reino Unido e Turin, na Itália. Os resultados esperados foram a redução consumo de energia e emissão de CO2, usando distribuições mais eficientes de energia [38].

2.2.8 Almanac

Segundo Bonino, Almanac é uma plataforma aberta que visa dar suporte a desenvolvimento de aplicações sustentáveis para cidades inteligentes, buscando integrar redes IoT com sistemas urbanos [24].

A plataforma possui vários tipos de interfaces para ajudar no desenvolvimento de aplicações, desde a interface de sensores de baixo nível e captura de dados, até o suporte de alto nível de processos, incluindo a integração de novos dispositivos e políticas da cidade [25].

Almanac coleta e analisa dados, podendo ser em tempo real ou não, de diversos sensores heterogêneos e atuadores dentro de uma cidade. O elemento chave dessa plataforma está no seu *middleware* baseado em arquitetura *Service-Oriented Architecture* (SOA), com foco na interoperabilidade dos recursos da plataforma. O *middelware* possui

uma rede que é construída dinamicamente, suportando vários aplicativos de terceiros para cidades inteligentes [24].

2.3 CONSIDERAÇÕES INICIAIS SOBRE AS PLATAFORMAS

As plataformas destacadas acima, foram pesquisadas com o objetivo de serem estudadas para o desenvolvimento de um projeto de cidades inteligentes. Destas plataformas, Interscity, SmartSantander, Dimmer, Almanac e Fiware possuem perfis que se assemelham ao problema de pesquisa proposto neste trabalho. Na sequência será destacada a arquitetura de cada uma destas plataformas.

2.3.1 InterSCity

A plataforma oferece serviços *REST*, de modo que seja possível interconectar diversos serviços e aplicativos para cidades inteligentes. A plataforma é constituída por seis microsserviços. *Resource Adaptor*,*Resource Cataloguer*, *Data Collector*, *Actuator Controller*, *Resource Discovery* e *Resource Viewer*.

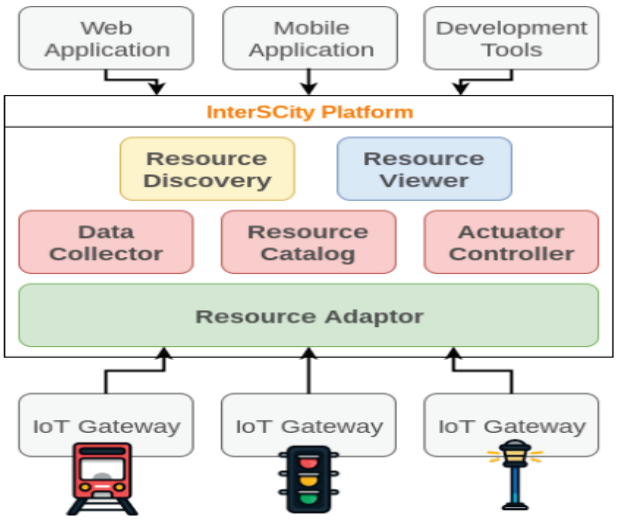

Figura 2. Arquitetura do InterSCity [20].

Na Figura 2 pode-se visualizar a arquitetura geral da plataforma InterSCity. O microsserviço *Resource Adaptor* é responsável por fornecer a integração entre os dispositivos IoT. *Resource Catalog*, *Data Collector*, e *Actuator Controller* cuidam da gerência dos dados. A visualização dos dados é feita pelo *Resource Viewer* [39].

- **Resource Adaptor** Essa aplicação atua como um adaptador de sensores para a plataforma, comunicando-se diretamente com os dispositivos físicos e encapsulando dados específicos dos sensores IoT para a plataforma, sendo possível simular sensores

ou atuadores nesta plataforma também. Nele são gerenciadas as interações com os dispositivos IoT, desde coletar os dados, armazená-los e se comunicar com sensores e atuadores, sendo que estes precisam ser registrados na plataforma antes de enviarem os dados.

- **Resource Catalog** Este microsserviço é responsável por armazenar e disponibilizar todos os dados disponíveis referente a um dispositivo IoT, como status, configuração e localização. Os dados ficam disponíveis para serem pesquisados por qualquer microsserviço através de mensagens assíncronas. Também é responsável por promover um *Universally Unique Identifier* (UUID) para cada sensor.

- **Data Collector** É responsável por armazenar os dados provenientes dos dispositivos IoT nas cidades, podendo ser eventos ou somente coleta periódica. Os dados podem ser pesquisados por meio de uma API e métodos de busca que a plataforma oferece, disponibilizando um conjunto de filtros para pesquisa de modo que consiga achar dados históricos mais facilmente.

- **Actuator Controller** Fornece serviços padronizados para intermediar as solicitações de atuadores da cidade. Além disso, esse controlador registra o histórico das solicitações de atuação para que elas possam ser acessadas no futuro.

- **Resource Discovery** É uma API que fornece um mecanismo de pesquisa e filtros para combinar informações, dados e metadados de forma que seja possível descobrir recursos ou eventos de uma determinada cidade.

- **Resource Viewer** É responsável por disponibilizar a visualização de dados, gerando gráfico com visões gerais e administrativas de uma cidade, incluindo localização, dados em tempo real e gráficos históricos com base nos dados dos microsserviços *Data Collector* e *Resource Catalog*.

A maior parte da comunicação dos serviços é feita via mensagens assíncronas usando o protocolo *Advanced Message Queuing Protocol* baseando-se no modelo *publish-subscribe* [40], embora tenham uma *API REST* que possibilite o envio de mensagens assíncronas por meio do protocolo de comunicação *HTTP* [20].

A plataforma InterSCity oferece uma amplo número de rotas para serem utilizadas. Cada microsserviço possui as suas definitivas rotas para requisições.

2.3.2 SmartSantander

A arquitetura do SmartSantander é composta por três nódos (redes) principais que são responsáveis pela coleta, transporte e mineração/processamento dos dados. A Figura

3 possui uma visão geral da arquitetura do SmartSantander, detalhando seus principais nodos.

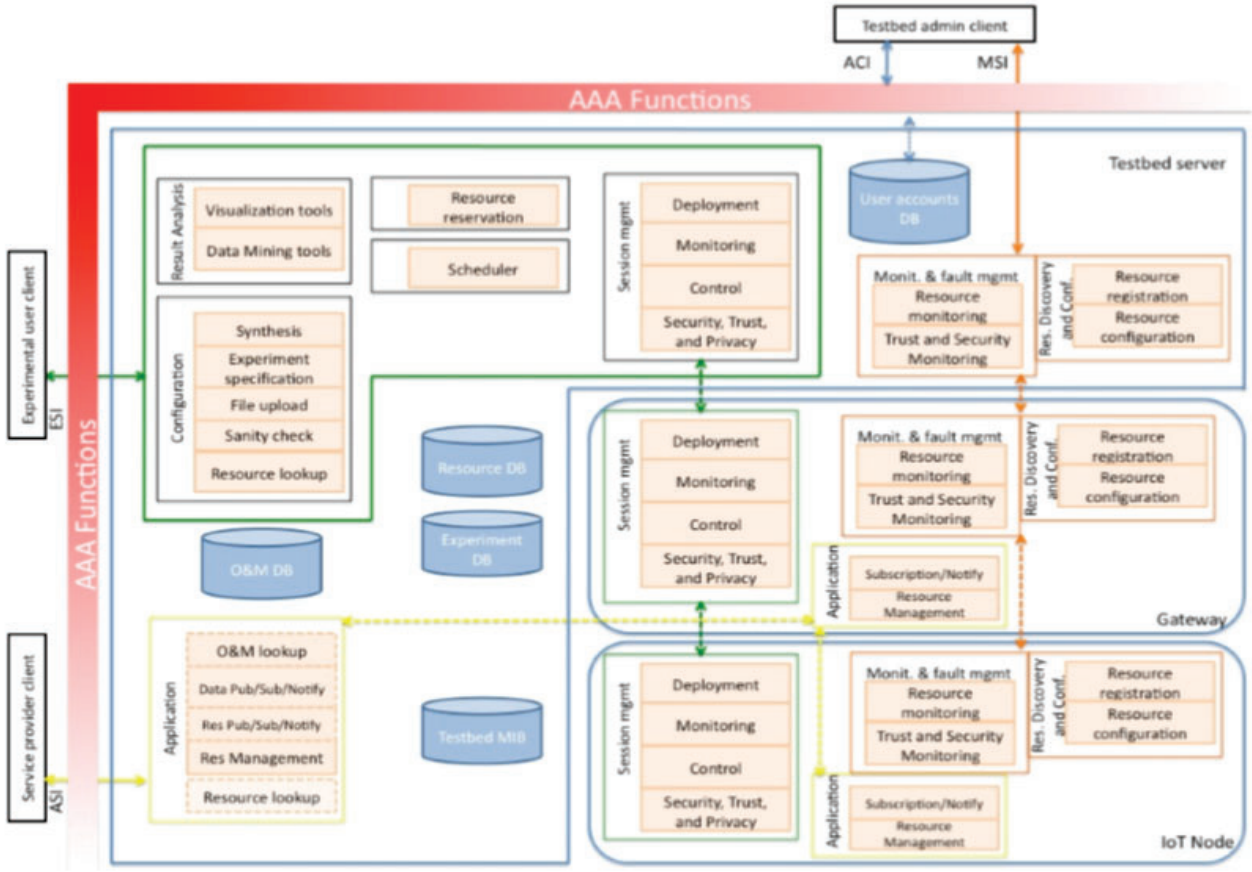

Figura 3. Arquitetura em alto nivel do SmartSantader [27].

Na Figura 3 pode-se visualizar os componentes da plataforma *SmartSantander*. Os componentes principais da arquitetura são *Device Tier*, *IoT Gateway* e *Server Tier*. *Device Tier* é formado pela rede de sensores que estão integrados na plataforma, onde coletam os mais variados tipos de dados. *IoT Gateway* é composta por um hardware com mais capacidade de processamento, onde recebe os dados captados pelo *Device Tier*, tratando-os e entregando para o *Server Tier*, que processa os dados, gerando informação para ser utilizada.

De acordo com Gutierrez, *Device Tier* é composto por vários tipos de sensores, geralmente equipados com RFID ou NFC, cujo o objetivo é captar e coletar dados. Geralmente ficam em lugares públicos, como nas ruas ou em postes de luz, expostos a qualquer tipo de evento climático, tais como chuva, temperaturas elevadas e baixas ou danos físicos [41].

Para minimizar os danos causados e focar na continuidade do serviço, estes dispositivos geralmente podem contar com dois tipos de fornecimento de energia, um contínuo, provenientes de postes ou tomadas e o outro via baterias. Possuem dois meios de comunicação com o *Gateway* para aumentar confiabilidade de que os dados chegarão ao destino.

Os *smartphones*, por terem uma capacidade de comunicação ampla e um poder computacional razoável, também são integrados neste nodo. Geralmente estes sensores são agrupados em *clusters*, onde dependem do nódo *IoT Gateway* [41].

IoT gateway é o nodo responsável por fazer a integração dos sensores IoT para a infraestrutura de rede, onde processam os dados. É necessário que estejam conectados a energia e que tenham um ponto de acesso à internet. Neste nodo é possível fazer o gerenciamento de comandos para os dispositivos, tendo controle de toda a rede.

O *hardware* de um nodo *gateway*, se necessário, é capaz de emular sensores virtuais com os mesmos aspectos de um sensor físico para continuar gerando dados para o fim desejado, esta função é usada para gerar dados para o *testbed* da plataforma.

O nodo *Server tier* é a infraestrutura de servidores utilizada para armazenar os dados que podem vir dos *Gateways* ou de outros serviços da rede, além de poder explorar técnicas de mineração de dados para geração de informação útil. Pode possuir mais serviços de acordo com a necessidade da aplicação, ficando disponíveis por meio de uma camada de virtualização.

A estrutura do *Server tier* é reprogramável, sendo possível trocar os métodos padrões de comunicação por outros, para implementar um serviço que a plataforma necessita. Os servidores que gerenciam redes diferentes podem se conectar entre si para trocar informações sobre os nodos ou dispositivos IoT que possuem.

O SmartSantander possui um subsistema de gerenciamento para *testbed*, onde possui funções que são relevantes para o gerenciamento dos testes na plataforma de modo que tenha um *testbed* contínuo. O sistema possui configurações prontas para novos componentes de testes, além de monitoramento de desempenho e gerenciamento de falhas [26].

2.3.3 Almanac

A arquitetura da Almanac foi projetada de modo que seja de fácil adaptabilidade em qualquer cidade que fosse instalado. Além de habilitar que aplicações externas acessem seus serviços, a plataforma possui um manuseio e comunicação de dados eficientes dentro da arquitetura [24].

A Figura 4 demonstra uma visão da arquitetura do projeto Almanac, sendo dividido em quatro camadas: *API*, *Virtualization*, *Data Management* e *Smart City Resource Adaptation*. A definição de cada camada está descrita na sequência:

- **API**: A camada API engloba módulos para integrar a plataforma com aplicações de terceiros, como a *API RESTFul* para interação cliente/servidor e um *websocket* para gerenciamento de controle de fluxo de dados.

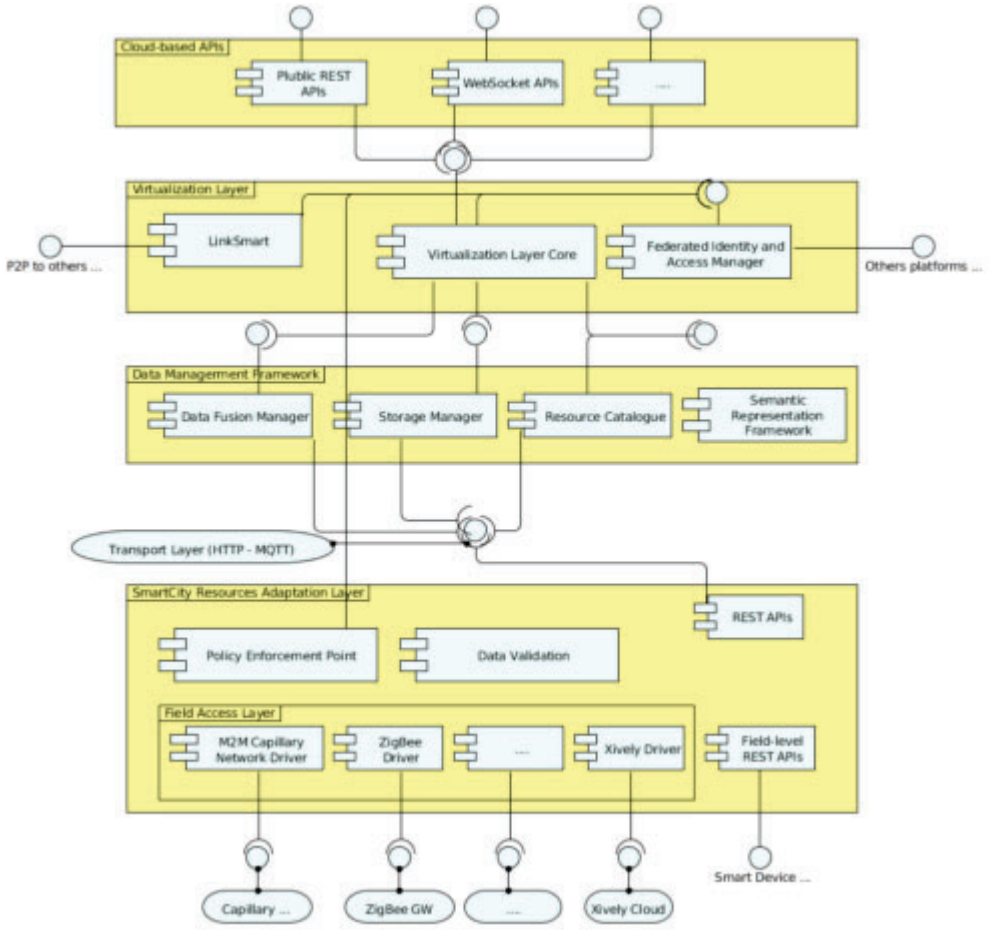

Figura 4. Arquitetura da Almanac [24].

- **Virtualization**: Essa camada é responsável por coordenar as atividades da plataforma, tais como: comunicação entre as camadas da plataforma, segurança dos dados e aplicação de políticas de acesso.

- **Data Management**: É a camada responsável pelo armazenamento, recuperação e gerenciamento dos dados coletados pela plataforma, possuindo submódulos que implementam técnicas de pesquisas eficientes, acelerando a busca por informações.

- **Smart City Resource Adaptation**: É responsável por fornecer acesso transparente aos dispositivos físicos que atuam na cidade, além de impor direitos de acesso e executar validação de dados recebidos.

Almanac possui uma estrutura na qual várias instâncias da plataforma podem participar de federações que oferecem serviços em diferentes cidades. Essas federações são desenvolvidas seguindo acordos de troca de serviços, geralmente por questões políticas. Essas instâncias podem trocar dados e distribuir funções de acordo com o privilegio que tem dentro da federação. Cada instância pode se conectar com mais de uma federação, podendo construir uma rede interconectada entre cidades [24].

2.3.4 Dimmer

A plataforma Dimmer surgiu para suprir necessidades de coleta de dados para *smart grids* em tempo real, além de ser *middleware* com tecnologias heterogêneas e comunicação assíncrona. Esta plataforma integra centenas de sensores e atuadores heterogêneos com o objetivo de gerenciar o consumo de energia. A plataforma possui dois sistemas de comunicação baseados em REST e *Message Queue Telemetry Transport* (MQTT) [23].

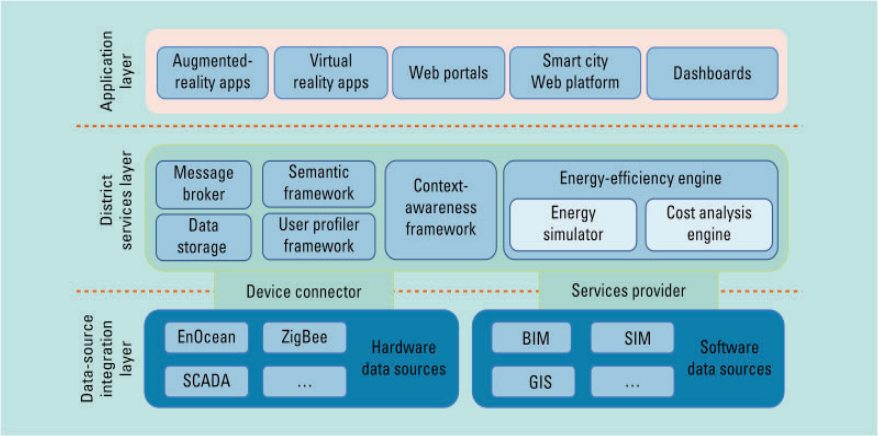

Figura 5. Arquitetura da plataforma Dimmer [22].

A Figura 5 representa uma visão da estrutura da plataforma. Os dados que são coletados através do componente *Data-Source integration Layer*. O processamento dos dados ocorre em *District Services Layer*, onde possui uma serie de componentes, cada um processando os dados para fins diferentes. Por fim, o componente *Application Layer* aplica ferramentas de pós-processamento para entregar a informação para o usuário final.

- **Data-Source integration Layer**: É responsável por manter a interoperabilidade entre as tecnologias existentes, tanto hardware quanto *software*. As tecnologias de baixo nível, como dispositivos IoT, possuem diferentes tipos de dados. Na plataforma, todos os dados recebidos são padronizados para um formato específico. Os dados são coletados de algumas fontes, tais como *Building Information Modeling* (BIM) um modelo paramétrico 3D para cada construção em uma área, *Geographic Information System* (GIS) fornecendo informações georreferenciadas de diferentes entidades em uma área, *System Information model* (SIM) representando modelos de rede de energia, além de sensores IoT espalhados por uma área especifica [22, 38].

- **District Services Layer**: Esse componente é o núcleo da arquitetura, sendo responsável pelo processamento dos dados. Existem vários subcomponentes que executam tarefas diferentes, de acordo com seu objetivo [22, 23].

  - **Message broker**: Responsável por fornecer uma comunicação assíncrona entre os componentes, usando o protocolo MQTT. Dessa forma, um componente pode enviar mensagem para os demais, sem a necessidade de aguardar a resposta.
  - **Data storage**: Tem o objetivo de gerenciar e armazenar dados provindos dos dispositivos citados acima. Também pode fazer integrações com banco de dados já existentes.
  - **Semantic framework**: Utiliza tecnologias de web Semântica para enriquecer os dados de uma área, usando metadados específicos sobre domínios e objetos envolvidos no edifício, rede de distribuição e níveis distritais. Estes dados então são armazenados no formato *Resource Description Framework*.
  - **Context-awareness framework**: Fornece modelagens reais usando dados de uma determinada área como entrada, criando Modelos e previsões de acordo com a entrada. Também é possível criar eventos para realizar determinada ação quando algum dispositivo muda seu estado.
  - **User-profiler framework**: Prevê comportamentos humanos de acordo com suas ações e *feedbacks* gerados. Estes comportamentos estão relacionados ao consumo de energia da área, disponibilizando possíveis melhorias no comportamento do usuário de modo que possa melhorar o consumo de energia.
  - **Energy-efficiency engine**: Responsável por explorar os dados dos outros componentes da plataforma ou em tempo real, podendo ser através dos modelos BIM ou SIM para simular políticas de optimização de energia e avaliar o impacto econômico que essa políticas terão, sugerindo um controle mais adequado nas tarifas de energia.

- **Application Layer**: Esse componente é responsável por gerenciar o pós-processamento de informações, através de um conjunto de APIs e ferramentas, entregando informações para o usuário. Novos aplicativos podem ser desenvolvidos para entregar novos tipos de informações provenientes dos dados dos componentes da plataforma. Na plataforma existem três níveis de dados. O nível de usuários inclui aplicativos para fornecer sugestões de otimização da energia aos usuários e coletar *feedback* para definir os padrões de comportamento, explicado no subcomponente *User-profiler framework*. O nível do edifício representa os aplicativos que fornecem informações ou sugestões sobre um edifício ou construção de uma área. O nível distrital inclui aplicações e ferramentas para monitorar e administrar toda uma área.

Os dados fornecidos pela plataforma Dimmer tem grande potencial para que aplicativos sejam desenvolvidos para profissionais que gerenciam edifícios ou bairros. Aplicativos para usuários comuns também podem ser desenvolvidos, visando a conscientização sobre o consumo de energia [22].

2.3.5 Fiware

A arquitetura do Fiware é constituída por APIs e *middlewares* distintos chamados *Generic Enablers*(GE), onde cada GE possui um objetivo específico [32]. Fiware possui uma grande quantidade de GEs que podem ser usadas para diversos tipos de desenvolvimentos, como por exemplo, interfaces de usuários, hospedagem na nuvem, gerenciamento de dados, segurança, entre outros.

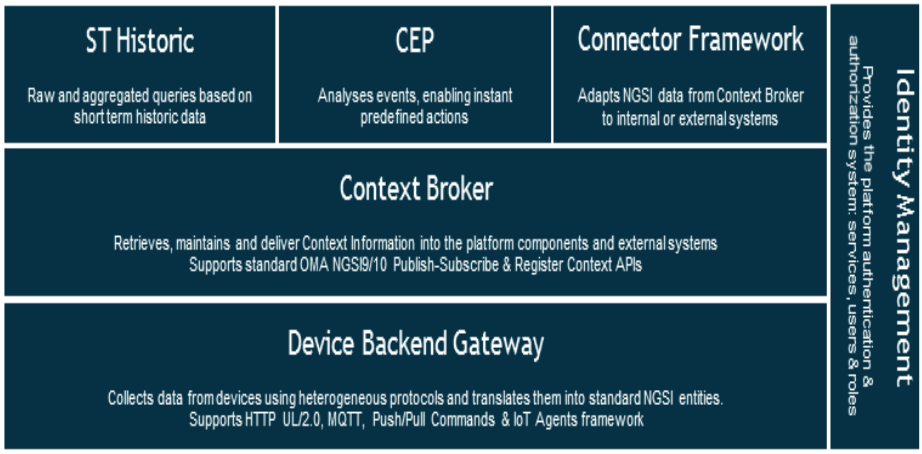

Figura 6. Componentes do Fiware [37].

A Figura 6 apresenta uma visão geral de alguns componentes do Fiware. Dentro de cada componente, existem vários GEs. Cada um destes, tem pelo menos uma especificação e uma implementação em código aberto para que os desenvolvedores possam usar como referência.

- **Device BackEnd Gateway** Conhecido como *Agent* IoT, é responsável por coletar os dados provindos dos mais variados objetos e sensores que usam protocolos heterogêneos, traduzindo-os para interface padrão da plataforma, chamada NGSI [42].

- **Context Broker** Esta estrutura recebe os dados do componente *Device BackEnd Gateway*, mantém e distribui em larga escala usando REST API para os todos os outros componentes da plataforma.

- **ST Historic** Este componente é responsável por armazenar de forma histórica os dados que chegam do *Context Broker*. Sendo assim, é possível recuperar informações antigas a respeito de um determinado item.

- **Connector Framework** É responsável por persistir os dados do componente acima, criando uma visão histórica dos dados para que possa ser acessado posteriormente por outro componente.

- **CEP(Complex Event Processing)** Os dados que são coletados no *Device BackEnd Gateway* são analisados em tempo real pelo CEP, podendo disparar algum evento específico caso o processo tenha encontrado alguma anomalia ou algum parâmetro fora do comum nos dados [37].

Na área da segurança, o Fiware dispõe de três GEs: *Identity Management*, PEP *Proxy* e *Authorization* PDP. O primeiro fornece mecanismos de autenticação para vários pontos dentro da plataforma, tais como usuários, atributos, serviços, entre outros. O Segundo é responsável por controlar o acesso aos recursos da plataforma, quando serviços de terceiros necessitam acessar algum recurso, o GE analisa o usuário para garantir que é autenticado. Este componente interage com os outros dois para verificar autenticação e autorização. O terceiro GE concentra-se na parte de autorização de solicitações. O GE PEP envia a solicitação para o PDP, que analisa e autoriza ou não a solicitação [43].

2.4 COMPARATIVO ENTRE PLATAFORMAS ESTUDADAS

Com o objetivo de comparar as plataformas estudadas, um conjunto de características foi selecionado, representando papéis fundamentais dentro de um ambiente de cidades inteligentes. Ao final, foi escolhida uma plataforma que servirá de base para este projeto de pesquisa. Os conceitos analisados na pesquisa foram formulados seguindo como base, os estudos de Tomas et. al. e Santana et. al [18, 19].

Uma característica que tem um contexto importante dentro de um ambiente de cidades inteligentes é o monitoramento de dados em tempo real. Em determinadas áreas, existe a necessidade de tomar decisões rápidas de acordo com determinado acontecimento. Além disso, essas informações analisadas em tempo real podem prever fenômenos ou ações futuras.

A interoperabilidade é uma questão muito discutida em projetos de IoT e cidades inteligentes. Existem uma variedade de objetos: sensores, atuadores ou qualquer objeto que consiga realizar algum tipo de processamento de dados, cada um deles com seus próprios métodos de comunicação e produção de dados. Portanto é necessário que esses objetos, mesmo com especificações técnicas diferentes, consigam comunicar-se uns aos outros, usando um meio padronizado de comunicação [44, 45].

A ubiquidade compreende que a tecnologia esteja presente em todo o cotidiano de uma cidade e seja capaz de captar dados sobre o ambiente em que está, como temperatura e umidade por exemplo. Esse conceito é importante para que o requisito de monitoramento em tempo real seja implementado [46].

A aplicação do conceito de ubiquidade produz uma grande quantidade de dados, um efeito chamado de Big Data [47, 48]. Com essa quantidade de dados a disposição, a segurança desses dados se torna um elemento importante para a proteção. Dependendo

da plataforma, os dados captados podem ser sensíveis, logo uma política de segurança dos dados tem sua importância [49, 50].

Aplicações de código aberto, possuem vantagens como por exemplo, possibilitar o entendimento do funcionamento da plataforma. Neste projeto, o acesso ao código fonte da plataforma é de grande importância, podendo moldar a plataforma para que atenda aos objetivos do projeto. Aplicações com seu código fonte fechado possuem limites de uso impostos pelos desenvolvedores do *software*, ficando a desejar quando a plataforma não oferece a funcionalidade que se necessita.

A arquitetura em camadas fornece maior estabilidade e independência para a plataforma, uma vez que pode separar as funcionalidades da plataforma por camadas, sendo cada uma responsável por realizar tarefas específicas, sendo capaz de alocar mais recurso caso seja necessário, sem interromper o funcionamento. Essa característica está diretamente ligada a um princípio das plataformas chamado Escalabilidade. As camadas podem ser separadas fisicamente ou não, mas o ideal é que a plataforma consiga trabalhar caso uma camada da arquitetura não esteja funcionando.

O suporte ao usuário é algo fundamental para qualquer ferramenta que presta algum tipo de serviço. Nos casos das plataformas, o suporte para desenvolvedores é importante afim de agilizar os processos de desenvolvimento com o esclarecimento de dúvidas técnicas. Para testar esse ponto, um pequeno *e-mail* contendo dúvidas a respeito da plataforma foi enviado para os suportes e para os respectivos responsáveis de cada plataforma. A resposta dessas dúvidas foi importante para escolha da plataforma.

A Tabela 3 contém uma relação entre as plataformas relacionadas com um conjunto de caracteriscticas que foram selecionados com base nas necessidades deste projeto.

Tabela 3. Comparativo das funcionalidades entre as plataformas analisadas.

Plataforma	InterSCity	Dimmer	Almanac	SmartSantander	CiVitas	Gambas	Kaa IoT	Fiware
Open-Source	x	x	x	x		x	x	x
Monitoramento em tempo real	x	x	x	x			x	x
Interoperabilidade	x	x	x		x		x	x
Ubiquidade	x	x		x			x	x
Segurança			x	x	x	x		x
Arquitetura	x			x			x	x
Suporte	x							x

A plataforma Dimmer difere-se das demais por possuir um objetivo específico. As outras plataformas tem como objetivo, dar suporte a aplicações de Cidades inteligentes com os mais variados temas. O Dimmer tem como objetivo o gerenciamento e otimização na utilização de energia elétrica em edifícios em áreas voltadas a *smart grids*. Mesmo se tratando de uma plataforma de código aberto e com um objetivo muito similar aos objetivos de pesquisa deste trabalho, o suporte ao desenvolvedor foi ignorado, sendo que o site oficial do projeto Dimmer, desde o momento de início até o momento de finalização da revisão das plataformas, estava sem acesso.

Nas plataformas Civitas e Gambas, não foram encontradas características como Monitoramento em tempo real e Ubiquidade. Funcionalidades como monitoramento de energia em tempo real por exemplo, seria algo à ser desenvolvido desde o começo, sem nenhuma base ou suporte das plataformas.

A arquitetura em camadas é um característica que foi analisada com o princípio de facilitar o entendimento na plataforma para a inclusão de um adaptador de serviço receptor de dados de energia elétrica. Neste ponto foi possível notar que plataformas como Civitas, Gambas, Dimmer e Almanac focaram na elaboração de uma arquitetura que atendia a objetivos específicos, dificultando a inclusão do adaptador.

O suporte é um ponto importante para o desenvolvimento de um *software* utilizando recursos de terceiros. As plataformas Almanac, Gambas, Civitas, Dimmer, e SmartSantander não responderam as dúvidas enviadas para seus respectivos suportes até o momento do fechamento deste trabalho e por isso, foram descartadas. Somente InterSCity e Fiware responderam os questionamentos enviados. Porém a equipe de suporte do InterSCity respondeu as perguntas com mais objetividade.

Na Tabela 3 nota-se que as plataformas InterSCity e Fiware possuem as características que foram julgadas necessárias para essa análise. Civitas e Gambas foram as que menos atenderam as expectativas. As características foram analisadas através das referências encontradas para cada plataforma.

InterSCity e Fiware possuem o mesmo objetivo de serem plataformas de código aberto padronizadas para aplicações de cidades inteligentes. Nas análises das características, é possível entender que às duas plataformas possuem as características necessárias e são mais adequadas para este trabalho.

Os processos de instalação das plataformas InterSCity e Fiware foram analisados. Ambas possuíam manuais de instalações com explicações de cada componente necessário. Para testar a instalação, foi usado um computador com sistema operacional Ubuntu 16.04, processador I7-3770, 8 GBs de memória e 500 Gbs de armazenamento. Ambas as plataformas foram instaladas com sucesso, sem erros durante a instalação.

A InterSCity chamou mais atenção e foi escolhida como plataforma base para este trabalho. Por possuir uma arquitetura de microsserviços, compreende mais facilmente a ideia de integrar um monitorador de energia elétrica residencial. Sua arquitetura é mais simples, não sendo necessário a utilização de um CEP, visto que todo o processamento será feito dentro do serviço de monitoramento. Além disso, o acesso ao suporte da aplicação mostrou-se mais objetivo na questão dos problemas enviados. A plataforma também possui autores brasileiros, podendo ser mais facilmente adaptada para projetos na realidade brasileira.

2.5 ISO 37120

A criação de novos métodos ou tecnologias que possam suprir necessidades ou resolver problemas das cidades precisam ser balizados por padrões existentes dentro de uma área. Estes padrões servem para nortear os desenvolvimentos, atuando como indicadores e limitadores de serviços. Tecnologias que seguem padrões estipulados pelo mercado, tem seus processos mais propensos a serem utilizados em uma cidade.

A ISO 37120 é a norma responsável pelo desenvolvimento sustentável de comunidades, disponibilizando indicadores para serviços urbanos e qualidade de vida. Foi criada com o objetivo de avaliar a sustentabilidade de uma cidade com base em indicadores de uma cidade sustentável [51].

A ISO possui uma série de seções distintas, cada uma com um contexto específico. Na seção de número 7 são apresentados os indicadores referentes ao consumo sustentável de energia elétrica. Seus indicadores de energia estão diretamente ligados aos objetivos deste trabalho e foram usados para ajudar na definição das informações importantes a serem disponibilizadas para o usuário. Os indicadores da seção 7 estão destacados abaixo:

- O consumo total de energia elétrica residencial per capita(seção 7.1);
- Porcentagem da população da cidade com serviço elétrico autorizado(seção 7.2);
- Consumo de energia de edifícios públicos por ano(seção 7.3);
- Porcentagem da energia total proveniente de fontes renováveis, como parte do consumo total de energia da cidade (seção 7.4);
- Uso total de energia elétrica per capita(seção 7.5);
- Número médio de interrupções elétricas por cliente por ano(seção 7.6);
- Duração média de interrupções elétricas(seção 7.7).

Os dados gerados através destes indicadores agrega ainda mais valor na visualização das informações, uma vez que os dados são tratados e disponibilizados conforme a indicação da ISO 37120.

3. MATERIAIS E MÉTODOS

Esta seção apresenta os materiais e métodos utilizados neste trabalho. O protótipo de monitoramento de eventos de energia elétrica é descrito sucintamente para entendimento do processo de detecção de um evento e os dados gerados do mesmo. O modelo de arquitetura proposto é destacado logo em seguida, visando um entendimento a respeito dos fluxos que o trabalho irá conter.

3.1 PROTÓTIPO DE MONITORAMENTO DE EVENTOS

O projeto desenvolvido por Mella, [52], tem a iniciativa de medir a energia elétrica de uma residência, com base em eventos, utilizando a metodologia do Protegemed [53]. O objetivo geral é avaliar uma adaptação dessa metodologia e verificar se este é capaz de mensurar o consumo de energia elétrica a partir dos eventos gerados pelas grandezas monitoradas, como a tensão e corrente. Para cada evento detectado, são salvos dez ciclos da rede dos quatro sensores que medem a tensão da rede, corrente de alimentação, corrente diferencial e corrente de aterramento, como demonstra a Figura 7. A aquisição e o processamento dos dados são realizados por um sistema embarcado composto por um microcontrolador *EK-TM4C1294XL* da *Texas Instrument* [54] que por fim codifica um pacote de dados, enviado para um servidor via rede *ethernet*. O hardware é baseado na última atualização do Protegemed, realizada em [53]. A Figura 7 mostra o esquema simplificado do hardware utilizado no projeto de Mella, M. J. [52].

Os sinais dos quatro sensores são digitalizados e processados no *firmware* do microcontrolador onde são realizadas as contagens das energias ativa, reativa e aparente além do tempo de contagem. Para cada ciclo da rede elétrica de 60Hz o sistema computa 256 amostras de cada sensor. Em seguida, do *firmware* realiza a análise dos quatro sinais para verificação de ocorrência ou não de eventos. Esses eventos são mudanças nos valores eficazes de um dos quatro sinais baseado no estado do último valor calculado e em valores predefinidos de limites máximo e mínimos. Um exemplo prático seria o da corrente de alimentação na qual inicialmente possui um valor de 1,0 *Arms* e ao ligar um aparelho passa a consumir uma corrente de 2,0 *Arms* sendo assim um evento chamado de *"EVENT_UP"* é gerado nessa mudança. Da mesma forma se esse aparelho fosse desligado e a corrente voltasse para o valor de 1,0 Arms, ocorreria um novo evento chamado de *"EVENT_DOWN"*. Os limites máximos e mínimos servem para garantir uma margem de intervalo para definir ou não a ocorrência de um novo evento. Os valores dos limites são estipulados com base em testes práticos. A mesma metodologia é adotada para os sinais de tensão da rede, corrente diferencial e corrente de aterramento [52].

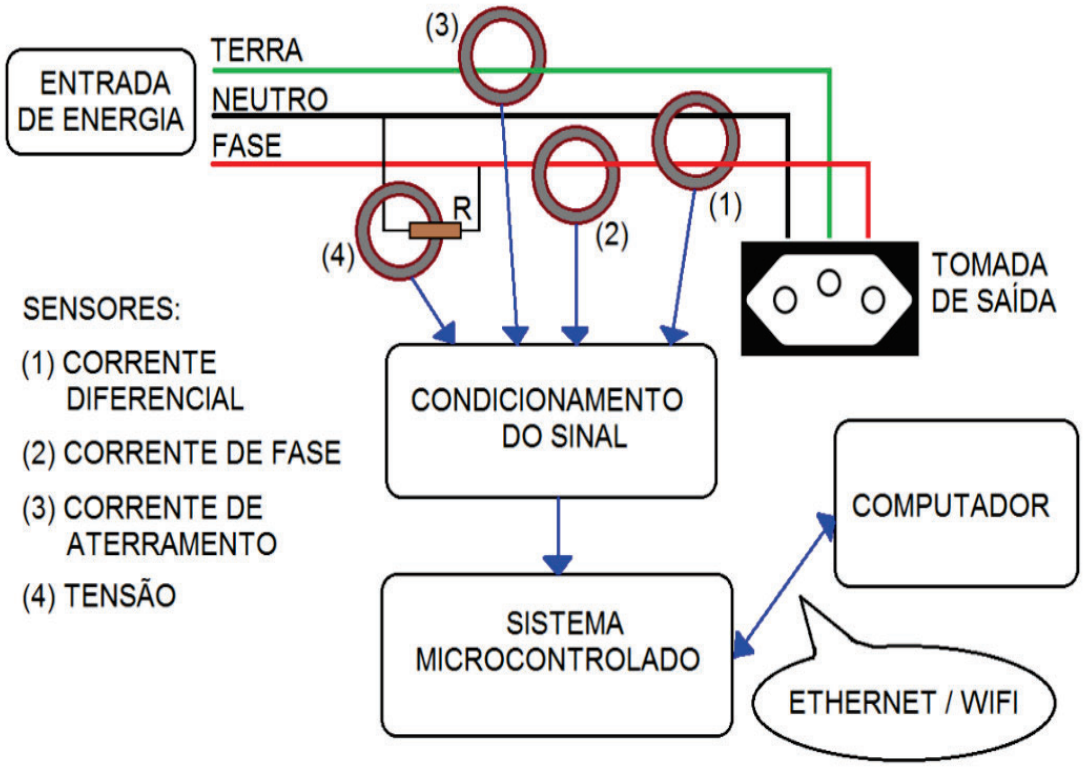

Figura 7. Arquitetura do dispositivo.

Para cada evento gerado são salvos dez ciclos de valores amostrados por cada um dos sensores, tornando possível uma análise detalhada sobre cada evento. Foi utilizado a interface de comunicação *ethernet* da placa *TM4C129* para envio dos dados e o protocolo de transporte *TCP/IP*. A organização e estruturação de um *array* de 15400 *bytes* foi dada da seguinte forma: Os primeiros 40 bytes contém os dados da contagem dos eventos, contagem das energias ativa, reativa e aparente, contagem do tempo, tipo de evento ocorrido, contagem de erros do *firmware* entre outros *bytes* reservados para futuras atualizações. Os demais 15360 bytes armazenam os dez ciclos de cada um dos quatro sensores.

Na Figura 8 é mostrada uma imagem do protótipo construído em [52]. Junto com este protótipo foi instalado um medidor eletrônico que serviu de referência para comparação e validação dos valores adquiridos nos testes de medição de energia.

No trabalho de Mella, M. J. foi implementado e validado a metodologia para medição do consumo de energia elétrica com base nos eventos gerados durante o período de análise estipulado nos testes. A contagem da energia elétrica no *firmware* é realizada em tempo real, isto é, ciclo a ciclo da rede e com as 256 amostras por ciclo da tensão e corrente sendo possível realizar o cálculo das potências ativa, reativa e aparente e com base no tempo de cada ciclo de 60Hz, aproximadamente 16,66 ms, o cálculo das energias. Variáveis em ponto flutuante são utilizadas para incrementar a contagem de cada energia assim como o tempo. A cada evento gerado essas variáveis são enviadas no *array* de

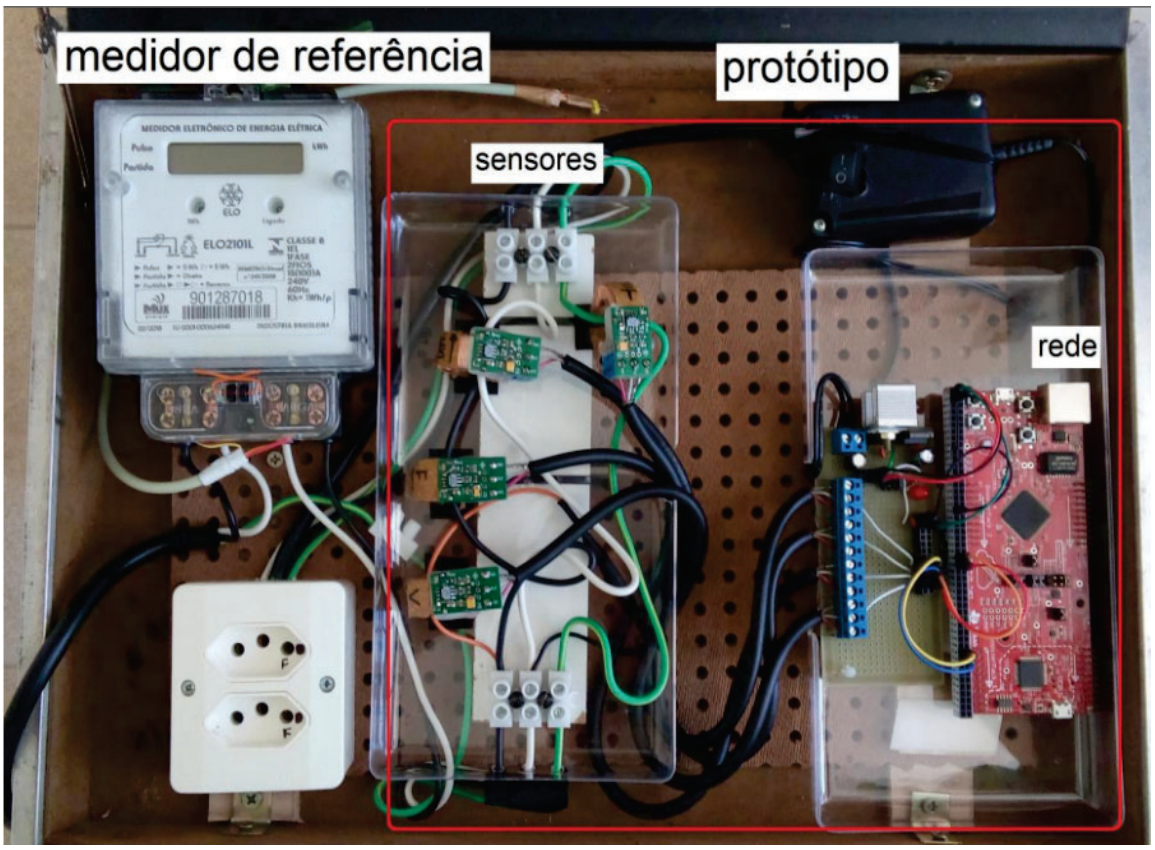

Figura 8. Protótipo implementado do dispositivo coletor de eventos de energia elétrica.

bytes para acompanhamento da contagem realizada no *firmware* e comparações com a contagem baseado nos eventos recebidos no servidor [52].

O resultado final é um banco de dados contendo os dados das principais atividades ocorridas no período de análise, servindo essas para fins de estudo de perfis de consumo, auditoria energética, desagregação de cargas, detecção de falhas e anomalias na rede elétrica causadas tanto pelo consumidor como concessionária entre outros e como encontrado no trabalho de AZZINI, H. A. D. [55]. Com base nos dez ciclos salvos de cada grandeza elétrica é possível também realizar para alguns aparelhos/cargas uma análise de transientes nos eventos de liga e desliga.

3.1.1 Dados gerados no protótipo

O trabalho relatado na seção 3.1, gera uma grande quantidade de dados que podem ser aproveitados para gerar várias informações úteis, tanto a nível de usuário, quanto a nível técnico, sendo base para trabalhos envolvendo cidades inteligentes com foco em energia. Os dados dos eventos são adicionados em um *Array* de 1540 *bytes* antes de enviar para o servidor. Para este trabalho, os dados gerados serão utilizados para obter informações de consumo e corrente elétrica em residências, com a possibilidade de detecção de alertas que podem ser prejudiciais para os equipamentos da residência.

Tabela 4. Principais dados gerados pelo protótipo.

Campo	Descrição
event	Tipo do evento(EVENT_UP ou EVENT_DOWN)
event_count	Contador de eventos dentro do protótipo.
energy_ativa	Consumo cumulativo dos eventos.
rms_Voltage_real	Tensão calculada do evento atual.
rms_Phase_real	Corrente calculada do evento atual.
rms_diff_real	Corrente de fuga calculada do evento.

Os dados projetados na Tabela 3.1.1 serão utilizados para este trabalho. Os dados brutos usados para o cálculo dos valores de tensão, corrente e consumo são também são salvos, porém são dados propícios a análises de características específicas da corrente elétrica sendo fornecida ou consumida, num nível de detalhe que extrapola os limites deste trabalho. Os mesmos são armazenados para um possível uso futuro em outras análises de dados.

3.2 ARQUITETURA DE MONITORAMENTO DA REDE ELÉTRICA EM CIDADES INTELIGENTES

O modelo de arquitetura desenvolvido foi focado em atender os esforços de processamento dos dados e no envio dessas informações para o InterSCity. Também deve ser responsável por processar qualquer outra funcionalidade que for implementada dentro do serviço.

O modelo de arquitetura, demonstrado na Figura 9, cumpre com os objetivos deste do trabalho, pois consegue integrar os três principais pilares envolvidos neste trabalho: *Hardware de monitoramento eletrico*, *InterSCity* e *dashboard*, tendo o serviço de processamento de dados como meio para descompactar e processar informações dos eventos.

O *dashboard* tem o objetivo de disponibilizar os principais dados dos eventos gerados através de gráficos e tabelas. Tem a responsabilidade de entregar ao usuário uma experiência de visualização de energia elétrica. O serviço de processamento de dados será responsável por descompactar os eventos e orquestrar as funcionalidades e métodos que exijam processamento de dados.

3.2.1 Funcionalidades

As funcionalidades descritas nesta seção foram projetadas para alcançar os objetivos deste trabalho. Começando pela descompactação de dados dos eventos, processamento dos alertas, processamento das funcionalidades, até o processamento de dados para visualização baseadas na ISO 37120 descrita na seção 2.5.

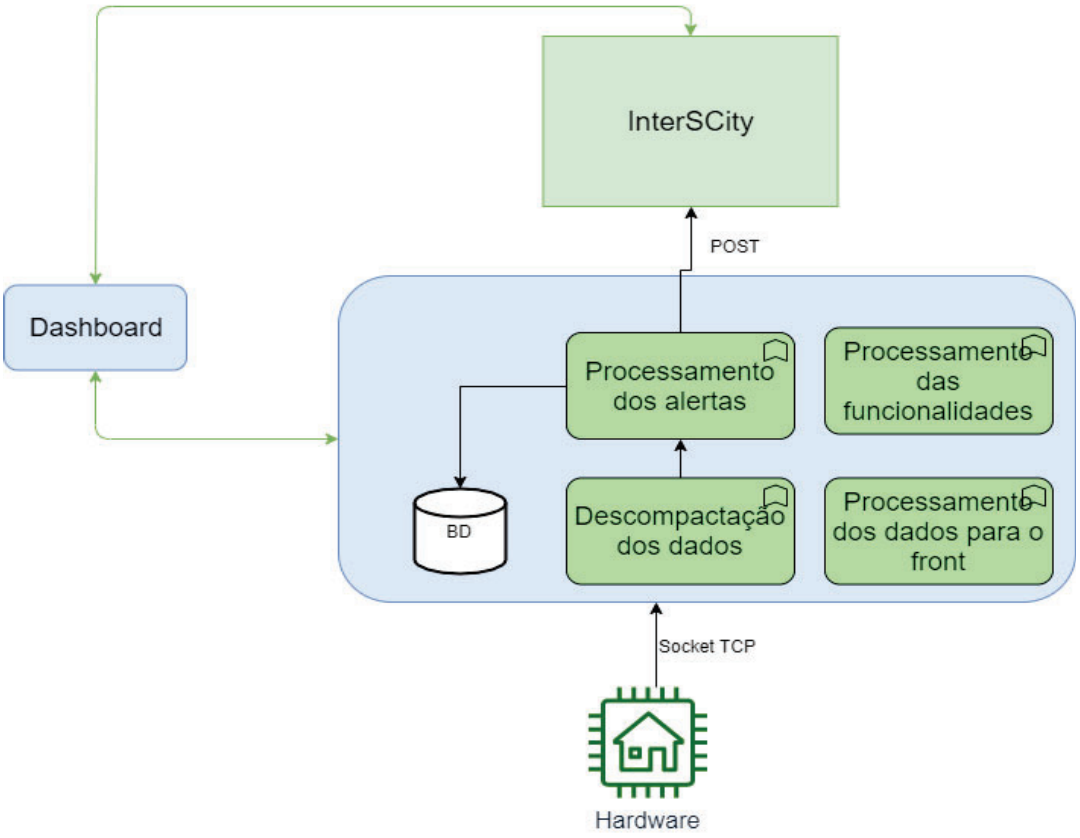

Figura 9. Modelo de arquitetura proposto para o comportar os objetivos do projeto.

Toda vez que houver uma nova residência a ser cadastrada no sistema, o serviço de processamento deverá receber os dados de identificação e cadastrá-la no InterSCity como um recurso. A partir deste momento, a plataforma retornará um UUID, que deve ser usado para identificação da residência nos eventos que serão enviados.

Os eventos gerados pelo protótipo de monitoramento de energia elétrica devem ser enviados para o serviço de processamento de dados, que por sua vez irá descompactar o evento e verificar se existe algum alerta. Após esta etapa estar concluída, os dados são enviados para o InterSCity, onde serão salvos para futuras pesquisas.

Os métodos para detectar eventos de alerta precisam ser processados primeiro, pois são responsáveis por indicar pra o usuário que a rede elétrica da residência está com algum defeito ou está passando por um momento crítico. Estes alertas podem ser de sobre-corrente (quando a corrente está acima do padrão nominal informado pelo usuário para a sua residência), sobretensão (quando a tensão está acima do nível informado), subtensão (tensão abaixo do esperado) e fuga de corrente indevida.

Outras funcionalidades de visualização devem ser processadas quando forem solicitadas, é o caso do informações a respeito do último evento, consulta de eventos em um determinado tempo, dados para disponibilização de um mapa e os indicadores da ISO 37120.

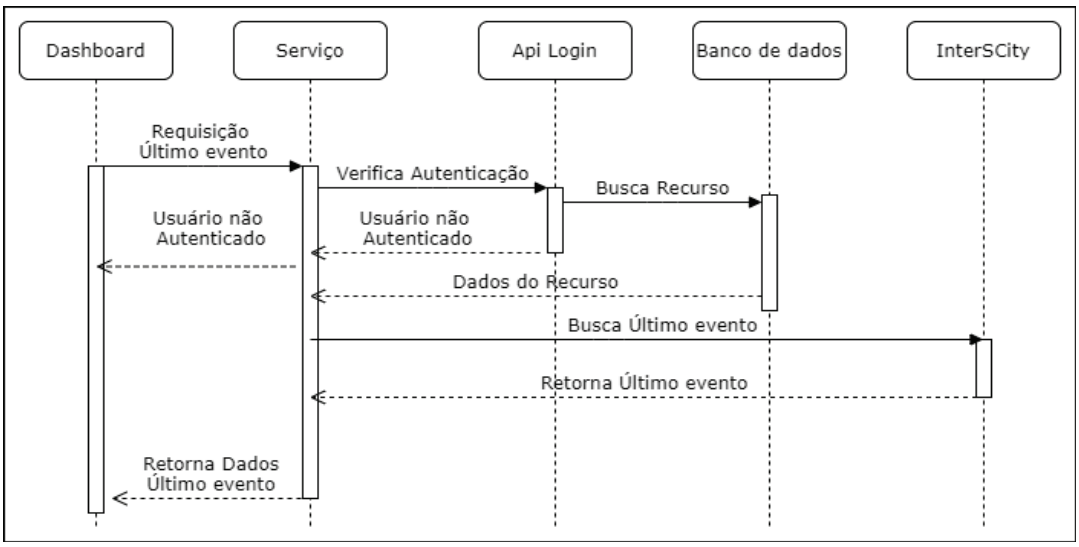

Figura 10. Diagrama de sequencia da busca do último evento.

O diagrama de sequência apresentado na Figura 10 é o projeto para visualização do último evento obtido pelo protótipo na residência. Quando o usuário informar o seus dados de identificação, o mesmo é enviado para o serviço de processamento, que valida a informação e busca o último evento no InterSCity. Os dados são processados e enviados para o *dashboard*, que os disponibiliza para visualização.

A busca por intervalo de datas possibilita que o usuário tenha visibilidade do seu consumo em determinado intervalo de tempo, podendo ajudá-lo a tomar decisões que diminuam o consumo elétrico na residência. Os dados do gráfico são processados no serviço, entregando os dados já formatados para o *dashboard*.

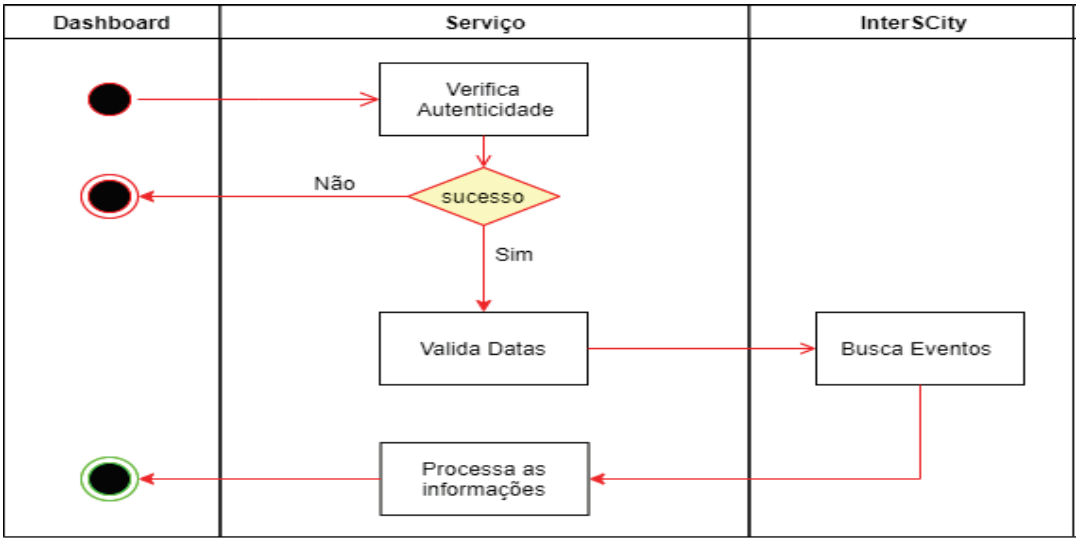

Figura 11. Diagrama de atividades para a busca de eventos por intervalos de datas.

O Fluxo descrito na Figura 11 relata os passos que são seguidos a fim de resgatar os eventos em um período de tempo para uma determinada residência. O usuário deve selecionar um intervalo de datas, que será repassado para o serviço de processamento.

Os eventos correspondentes ao intervalo das datas são resgatados e processados para gerar informações do *dashboard*.

As informações mostradas no *dashboard* para eventos em um intervalo de datas podem ser modificadas pelo usuário. Os modelos inicialmente projetados para o serviço serão baseados em informação de tensão, corrente ou consumo dos eventos. Todas devem ser geradas das informações dos eventos buscados no intervalo entre as datas recebidas.

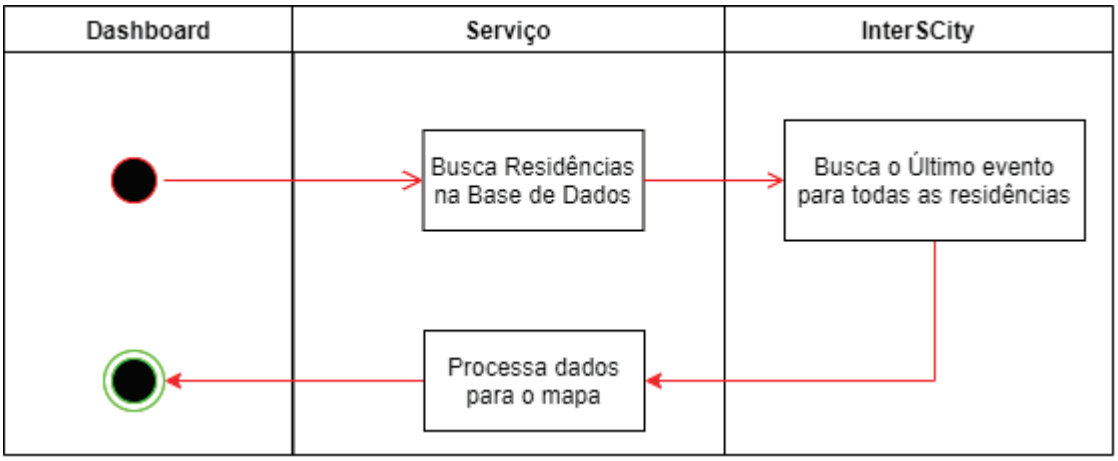

Figura 12. Diagrama de fluxos para o mapa de eventos.

O fluxo descrito na Figura 12 relata os passos para gerar os dados que são disponibilizados em um mapa. Ao acessar a tela, uma requisição é disparada para o serviço, que por sua vez, resgata todos os recursos que estão no banco de dados. Em seguida é resgatado o último evento de cada residência, gerando os dados necessários para a criação do mapa.

Os dados devem ser gravados já com indicação se houve alertas ou não, facilitando o serviço para endereçar os alertas, caso seja necessário. Para validar uma visualização de alertas, o mapa deverá comportar as principais informações dos eventos descritos na seção 3.1.1.

Os indicadores contidos na ISO 37120, destacados na seção 2.5, serão disponibilizados juntamente com o *dashboard* do mapa. Somente o indicador referente a seção 7.5 que será disponibilizado no *dashboard* de último evento, pois seus dados limitam-se a de uma única residência.

Para manter uma atualização constante dos dados no mapa, o serviço deve refazer o processo de consulta nos eventos das residências em certos períodos de tempo, recalculando também as funcionalidades que serão mostradas na interface juntamente com o mapa.

4. IMPLEMENTAÇÃO

A seção 4 descreve as ferramentas utilizadas para o desenvolvimento deste trabalho, bem como as funcionalidades implementadas no serviço e no *dashboard*. Por fim, é destacado a implantação de uma versão da plataforma InterSCity no Campus 1 da UPF, usada desde então para os testes.

4.1 RECURSOS UTILIZADOS

Inicialmente, a linguagem escolhida para a codificação do serviço foi o Ruby, na versão 2.7 [56]. A escolha foi motivada pelo uso da mesma linguagem utilizada na InterSCity para o desenvolvimento dos microsserviços, uma vez que estaria ambientado com as tecnologias empregadas no mesmo.

Porém, devido a constatação de que não seria necessário alterar o código-fonte da plataforma InterSCity, não era mais necessário utilizar a mesma linguagem para desenvolver o serviço para o tratamento dos dados de energia. Pensando na rápida curva de aprendizado e na utilização de bibliotecas disponíveis na internet, a linguagem adotada como padrão para o desenvolvimento do serviço foi o Python, na versão 3.6 [57].

A ideia de utilizar Python como linguagem principal também foi motivada pela biblioteca Flask, que oferece ferramentas para a criação de serviços com comunicação *REST*, bem como outras formas de comunicação disponíveis [58].

O rápido aprendizado da linguagem Python, bem como a vasta disponibilidade de materiais de estudo disponível na internet, também ajudaram na escolha da mesma, uma vez que o principal objetivo do serviço é orquestrar chamadas *REST* para buscar e processar dados.

Os testes com geração de gráficos e descompactação dos eventos, realizados no trabalho de Mella M. J. foram desenvolvidos utilizando a linguagem Python. Este motivo ajudou na escolha da linguagem [52].

O desenvolvimento do *dashboard* foi iniciado usando a ferramenta *Node-red*, na versão 1.0.3. O trabalho de Souza F. V. M. utiliza a mesma ferramenta para visualização dos dados coletados de dispositivos espalhados no Campus da UPF, para validar a sua arquitetura *LoRaWan* [59]. Para validar o objetivo do trabalho, era necessário que a ferramenta disponibilizasse meios de gerar *dashboards* de forma simples e intuitiva. Se trata de *framework* de desenvolvimento em blocos, onde cada bloco pode executar uma função, seja pré-programada ou dinâmica, seguindo um fluxo predeterminado pelo desenvolvedor. Os blocos são chamados de nodos, podendo possuir um ou mais blocos dentro de um determinado nodo [60].

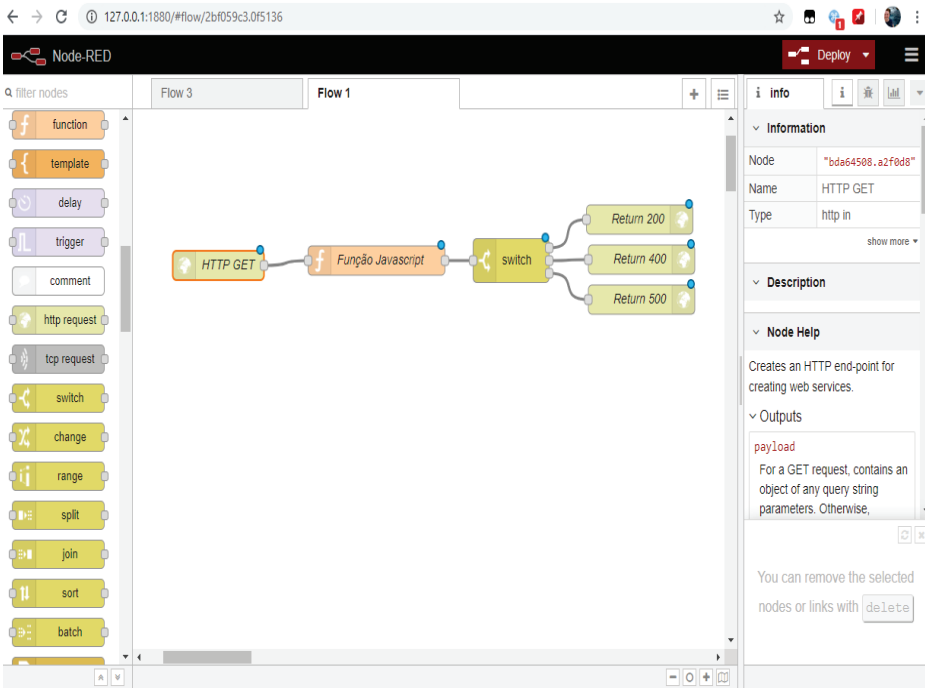

Figura 13. Ambiente de desenvolvimento do *Node-red*, construindo uma requisição *Rest*.

A Figura 13 mostra o ambiente de desenvolvimento do *Node-red*, onde foi gerada uma requisição *Rest*. Dependendo do conjunto de dados de entrada, a requisição pode retornar um status específico. Cada nodo possui uma função fixa, sendo executado somente quando o outro nodo terminar, seguindo um fluxo sequencial.

Durante a pesquisa dos nodos que o *Node-red* possui, foi encontrado um específico chamado *node-red-contrib-dashboard*. Este nodo fornece ferramentas prontas para uso de gráficos, calibradores, tabelas, entre outros. Por não precisar de configurações, é necessário somente ajustar os dados no formato correto de entrada. Este detalhe contribuiu para a escolha do mesmo como sendo a principal biblioteca a ser usada na construção do *dashborad* [61].

Para comportar a funcionalidade de demonstração de um conjunto de residências em um mapa, foi utilizado inicialmente a extensão *node-red-contrib-maps*. Os nodos disponíveis nesta extensão dispõem de ferramentas para a marcação de locais específicos no mapa, bastando enviar um conjunto de informações a respeito da residência que o nodo disponibiliza um ícone na localização enviada, com uma pequena descrição sobre o mesmo e uma informação que é de escolha do desenvolvedor [62]. Porém, após uma sequência de testes, a biblioteca parou de funcionar repentinamente. Devido a isso, foi iniciado uma busca por uma nova biblioteca que oferecesse ferramentas semelhantes.

Após uma extensa busca nas comunidades do *node-red*, não foi encontrado nenhuma biblioteca semelhante que oferecesse uma integração de mapas. Por tanto, foi decidido que seria utilizado as APIs do *Google Maps*, disponibilizadas pela *Google*. Essas APIs

não são integradas com o *Node-red*, necessitando a criação de uma página específica para integrar o mapa [63].

A API do *Google Map*, necessita de cadastro e é parcialmente gratuita, a fidelidade inclui um número de requisições que podem ser feitos de forma gratuita. Após o seu uso por completo, o produto passa a ser cobrado financeiramente. Para o desenvolvimento deste trabalho, a quantidade de requisições gratuitas eram suficientes, não necessitando contratar planos pagos [63].

Durante a utilização do *Node-red* no desenvolvimento das telas, notou-se uma grande dificuldade em utilizar as bibliotecas do *node-red*. As documentações eram, muitas vezes, desatualizadas e o formato dos dados não condiziam com o que a ferramenta realmente esperava. Houve dificuldades na personalização das bibliotecas, uma vez que as ferramentas são muito limitadas e não permitem edições.

A etapa de implementação começou com o desenvolvimento de um serviço teste com uma interface *REST* e duas rotas *GET* e *POST* disponíveis para uso. O objetivo inicial era ambientar-se na ferramenta *Flask* e na linguagem *Python*, de forma a facilitar futuros desenvolvimentos.

Na InterSCity, todos os sensores que se comunicam com a plataforma devem obrigatoriamente estarem cadastrados na mesma. Pois o cadastro gera um UUID que será utilizado pelo sensor como forma de identificação para enviar os dados. Os sensores cadastrados na plataforma são chamados de Recursos [39].

Para cadastrar uma residência no trabalho, algumas informações devem ser especificadas para que sejam enviadas para a InterSCity. No contexto desse trabalho, um recurso será um protótipo de monitoramento de energia em uma residência.

Para comportar os dados dos eventos na InterSCity, foi criado uma capacidade chamada "infoConsumo". As capacidades são formas de identificar quais tipos de dados que o recurso pode enviar para a plataforma. Ao cadastrar um novo recurso, é necessário especificar quais as capacidades deverá conter, como forma de organizar os dados que o recurso pode enviar [39].

O armazenamento dos dados na InterSCity ocorre em um banco de dados não relacional. Por tanto, os dados não precisam estar estruturados, bastando que sejam enviados no formato *JSON* para a plataforma. As informações do evento detectado pelo protótipo de monitoramento de energia elétrica, descritos na seção 3.1.1, serão enviadas para a InterSCity no formato especificado.

4.2 DESCOMPACTAÇÃO E PROCESSAMENTO DOS DADOS

O primeiro passo para a construção do projeto era integrar o hardware de monitoramento de energia elétrica com um serviço. Por já utilizar o método de comunicação *Socket TCP*, foi necessário que o serviço também se comunicasse através deste protocolo.

Este serviço é focado em disponibilizar uma conexão *socket TCP* na porta 7891. Esta porta foi escolhida pois o *client* criado no hardware de monitoramento de energia elétrica, utilizava esta mesma porta para enviar os eventos. Como o objetivo deste trabalho não é focado na alteração do hardware, foi optado por manter a porta já configurada, visto que não teria problemas de concorrência de portas.

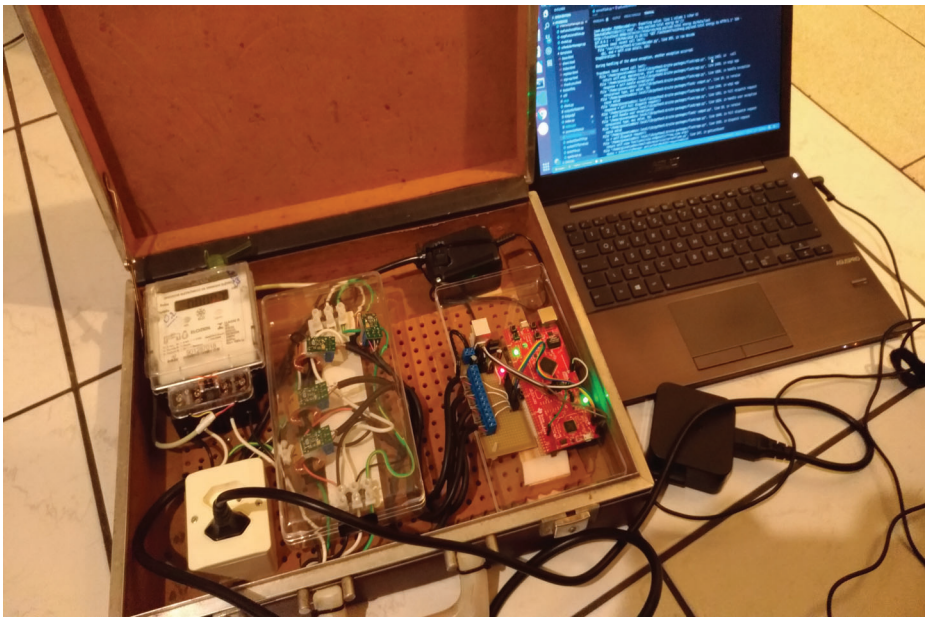

Figura 14. Protótipo de monitoramento de energia elétrica com o serviço de recebimento dos eventos.

Após o desenvolvimento do serviço, foi realizado o teste para conferir se o mesmo estava recebendo corretamente os dados do hardware. A Figura 14 mostra seu funcionamento, onde um carregador de *notebook* foi ligado ao protótipo de monitoramento de energia elétrica, que detectou e enviou eventos. O serviço recebeu a requisição do hardware com sucesso, validando esta etapa do projeto.

Durante o desenvolvimento do serviço para a recepção dos dados, notou-se a necessidade de identificar o hardware, pois as informações que eram enviadas não possuíam um endereço que representasse o mesmo. Pensando nisso, foi adicionado no *Array* de *bytes*, o UUID gerado no cadastro da residência na InterSCity. Dessa forma, é possível identificar qual é a residência que está enviando os dados para o serviço.

As dificuldades para o desenvolvimento desta etapa estavam voltadas para o entendimento do formato dos dados que eram gerados no hardware de monitoramento elé-

trico. Uma vez que o *array* de 1540 Bytes continha todos os dados compactados e precisavam ser descompactados dentro do serviço.

Além do entendimento do formato dos dados enviado do hardware, foi necessário entender como era descompactado o *array* para obter os dados originais. Como o serviço necessitava descompactá-los, foram utilizados os principais métodos de compactação e descompactação desenvolvidos no trabalho de Mella M. J. [52]. Desta forma, o processo de compactação e descompactação dos dados ficou no mesmo padrão do protótipo.

Com a consolidação do serviço de recebimento de eventos, o projeto já está apto a receber os dados do hardware de monitoramento de energia elétrica e descompactá-lo para que seja possível armazená-los. Para isso, foi criada uma rota para que os dados descompactados fossem enviados para a InterSCity.

A criação da rota para armazenamento dos dados na plataforma foi feita usando o padrão *RESTful*, com uma mensagem em formato *POST* contendo os dados no corpo da requisição em formato *Json*. Ao receber a requisição, a plataforma InterSCity retorna uma mensagem sinalizando que os dados foram salvos com sucesso para aquele recurso

Para casos em que a plataforma retornava algum erro ao armazenar os dados, estes não eram salvos e o evento é perdido. Essa ação foi tomada, visando o trabalho e o processamento que deveria ser criado para reenviar as mensagens em caso de falha.

4.3 BANCO DE DADOS

Durante o desenvolvimento do serviço, optou-se por adotar um banco de dados relacional para intermediar informações dos eventos com mais eficiência. Pensando nisso, o banco escolhido foi o *Mysql* [64]. A sua escolha foi definida pela larga escala de aplicações que utilizam este software e por ser gratuito. A dificuldade de saber qual era o último evento de uma residência foi o ponto focal para a criação de uma estrutura de banco de dados neste projeto. A inclusão da identificação do último evento não foi implementada no controlador pois seria necessário fazer o uso da memoria do microcontrolador, algo que geraria mais processamento e a possibilidade de dessincronizações. Para facilitar ainda mais a busca de informações e diminuir os acessos a plataforma InterSCity, foi criada uma tabela para guardar informações a respeito da residência.

A procura pelo último evento de uma residência específica era complexa, pois não existia um identificador relacionado ao evento. Desta forma foi criada uma tabela para orquestração dos IDs destes eventos, juntamente com outras informações.

A tabela *casa_info* guarda informações a respeito das residências cadastradas no projeto. Nesta tabela, armazena-se dados da residência, como UUID, corrente nominal, tensão nominal, validação de prédio público, verificação de produção de energia renovável, latitude e longitude, etc. A tabela *last_event* guarda as informações do último evento, como

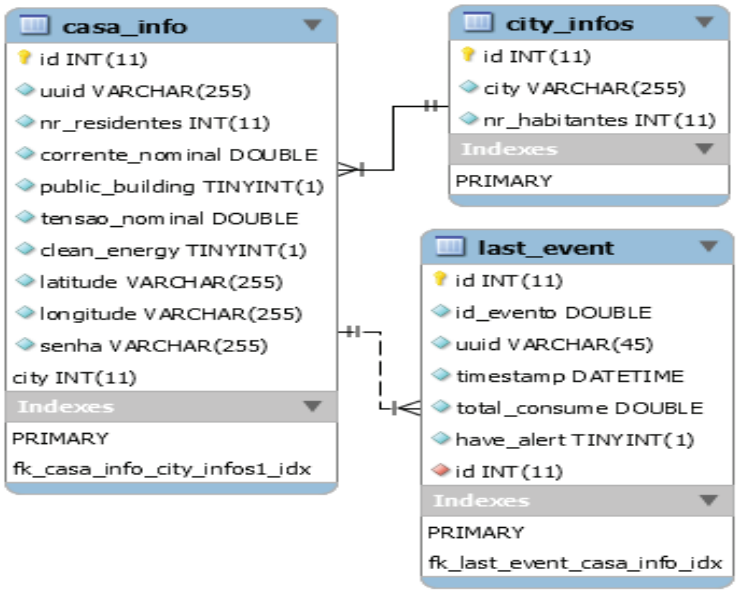

Figura 15. Diagrama do Banco de dados criado.

UUID da residência, Id do último evento e se houve alerta. Para vias de comparação, é necessário que seja mantido o id do evento tanto na tabela relacional, quanto na InterSCity. A tabela *city_info* contém informações das cidades que possuem os equipamentos de monitoramento. Nesta tabela é salvo o nome e o número de habitantes.

4.4 FUNÇÕES DISPONIBILIZADAS PARA O DASHBOARD

Antes de iniciar o desenvolvimento do *dashboard*, foi necessário criar um serviço que provesse uma estrutura orquestradora de chamadas entre o *dashboard* e a InterSCity. A decisão de processar os dados em um serviço *back-end* foi necessária devido a limitação dos *browsers* com processamento de grandes quantidades de dados via *javascript*. Porém, não existe um impeditivo em que baseando-se no UUID escolhido, o *browser* poderia requisitar os dados para a plataforma por meio de chamadas Ajax [65].

4.4.1 Funções da ISO 37120

Com a estrutura do serviço de recebimento e processamento de eventos concluída, iniciou-se o desenvolvimento das funcionalidades descritas na seção 2.5 referente a ISO 37120. Elas foram desenvolvidas separadamente para possibilitar a consulta específica em algum trabalho futuro que venha a utilizar o serviço.

A seção 7.1 da ISO, responsável por exibir o consumo total de energia residencial per capita foi construída possibilitando dois níveis de resposta. O primeiro é para aplicar a funcionalidade somente para uma residência específica, neste caso é necessário informar o

UUID do protótipo de monitoramento de energia elétrica, retornando o consumo per capita da residência. O serviço busca todos os eventos do *UUID* na InterSCity. O resultado é obtido através de somatório do consumo dos eventos, dividido pelo número de residentes do *UUID* específico. O segundo nível engloba o cálculo para todas as residências cadastradas, não necessitando informar nenhum dado. Este nível está diretamente ligado a funcionalidade da seção 7.5 da ISO, responsável pelo uso total de energia elétrica per capita.

Para calcular a porcentagem da população de uma cidade com serviço elétrico autorizado, descrito na seção 7.2 da ISO, é necessário informar qual a cidade que deseja realizar a consulta. O serviço buscará todas as residências da cidade escolhida no banco de dados, somando todos os residentes e dividindo pelo número de habitantes da cidade. Com estes dados é possível calcular a porcentagem de da população da cidade utilizando o protótipo de monitoramento de energia elétrica.

A funcionalidade de consumo de energia em edifícios públicos por ano(seção 7.3 da ISO) foi desenvolvida buscando todos os protótipos de monitoramento elétrico do banco de dados que identificam-se como prédios públicos, verificados através do campo *public_building* presente na tabela *casa_info*. Em cada uma destas residências busca-se todos os eventos do ano atual, obtendo o consumo total do edifício. Esta funcionalidade não é restrita ao ano atual, cabe ao usuário informar o ano desejado.

A seção 7.4 da ISO, responsável por medir a porcentagem de energia proveniente de fontes renováveis não foi implementada, pois o protótipo de monitoramento usado não possui leitura para este tipo de energia. Para contemplar essa seção, foi criado um campo novo no banco de dados, onde é possível saber se a residência utiliza alguma fonte renovável de energia.

O número médio de interrupções elétricas por cliente, destacado na seção 7.6 da ISO, é calculado com base no somatório de todos os eventos de marco zero dos protótipos de monitoramento de energia elétrica. Estes eventos são identificados através da verificação de um campo que mostra a quantidade de eventos que foram detectados enquanto a casa estiver com energia. O contador é zerado toda vez que o fornecimento de energia é interrompido. Nesta função, é possível calcular com todas as residências ou somente uma.

A última funcionalidade a ser desenvolvida foi o cálculo de duração média das interrupções elétricas que foram captadas pelos eventos(seção 7.7 da ISO). Essa função pode ser requisitada para uma residência em específico ou para todas. Ao buscar os eventos de marco zero na InterSCity, é analisado o ID do evento e em seguida, buscado o evento que antecedeu o mesmo, desta forma, é possível calcular o período em que a residência ficou sem energia elétrica.

4.4.2 Funções para as Residências Monitoradas

A partir da conclusão do desenvolvimento das funcionalidades da ISO 37120, o projeto passou para a etapa de criação de rotas que serão expostas para o *dashboard*. Por padrão, optou-se por transformar as respostas das rotas obrigatoriamente em formato JSON, para padronizar a leitura das informações em qualquer outro serviço que venha a utilizar as rotas criadas. A primeira a ser desenvolvida foi assinatura */lastevent* em formato *GET*, responsável por resgatar o último evento de uma residência específica.

Ao receber uma requisição para rota */lastevent*, a função busca o último evento gerado da residência através do seu UUID na plataforma InterSCity. A Figura 16 mostra a resposta gerado pelo serviço.


```
{
    "datajson" : {
        "event_type": "EVENT_UP",
        "energy_ativa": 0.4,
        "voltage_real_rms": 220,
        "phase_real_rms": 0.03,
        "alert_type": "none",
        "total_energy_daily": 2.2,
        "total_energy_percapita": 1.1
    }
}
```


Figura 16. Json gerado com dados do último evento.

A Figura 16 mostra o retorno da função com os dados do último evento. É composto por informações de consumo, voltagem, corrente, tipo do evento e o tipo do alerta. Para finalizar, é feito uma nova consulta na InterSCity a fim de buscar o consumo total diário e o indicador de sustentabilidade responsável pelo consumo total residencial per capita, descrito na seção 2.5.

Com o intuito de facilitar o cadastro das residências, foi criado uma rota que atende pela assinatura */register*. Os dados necessários para cadastro preenchem a tabela *casa_info*, como demonstrado na Figura 17.

Na Figura 17 encontra-se um exemplo das informações necessárias para cadastro da residência no sistema. Ao receber uma requisição na rota */register*, os dados da residência são enviados para a InterSCity, cadastrando um novo recurso e consequentemente, obtendo o UUID da residência. Após esta etapa estar concluída, os dados são salvos no Banco de dados.

Para disponibilizar as informações das residências no mapa do dashboard, foi desenvolvida a rota com assinatura "*/initialMap*". Quando uma requisição é recebida pelo

```
{
    "cpf": 03225415365,
    "first_name": "Giovani",
    "last_name": "Rizzardi",
    "nr_residentes": 2,
    "corrente_nominal": 14,
    "tensao_nominal": 220,
    "public_building": false,
    "latitude": "-28.26278",
    "longitude": "-52.40667",
    "cidade": 1
}
```


Figura 17. Json gerado com dados de um registro de uma residência.

serviço, todas as entidades das casas são recuperadas do Banco de dados. Em cada uma das entidades é buscado o último evento na plataforma InterSCity. Na Figura 18 é possível observar o formato de saída dos principais dados do evento para cada uma das residências.


```
{
    "datajson" : [
                {
                    "uuid": "432f.4234h.423fj4.555f",
                    "event_type": "EVENT_UP",
                    "energy_ativa": "0.9",
                    "voltage_real_rms": "234",
                    "phase_real_rms": "0.25",
                    "lat": -28.26278,
                    "lon": -52.40667,
                    "alert_info": "Sobretensão"
                }
            ]
}
```


Figura 18. Json gerado com dados para a geração do mapa.

Outra funcionalidade disponibilizada através da assinatura */attdashboard*, busca e processa eventos baseados em um intervalo de datas. Os dados da resposta também podem ser alterados, bastando que o tipo dos dados retornados sejam enviados. Ao enviar uma requisição contendo o UUID, data de início das buscas, data de fim e o tipo de dados que é necessário retornar(tensão, corrente ou consumo), é feita uma busca dos eventos na InterSCity, respeitando o intervalo de datas enviadas, sendo retornado o histórico dos valores baseado no tipo de retorno escolhido.

As funcionalidades da ISO 37120 estão disponíveis para consulta através da rota */dashboard1*. Ao enviar uma requisição para o serviço, todas as seis funcionalidades são executadas. Retornando o valor de cada uma para o destinatário. Esta rota deve ser utilizada para a tela que possui as informações do mapa, pois não possui nenhum UUID específico para pesquisas, uma vez que o objetivo é entregar uma visão geral de todas as funcionalidades. A Figura 19 contém um exemplo de retorno das funcionalidades.

```
{
    "response" : {
        "s72": 0.5835,
        "s73": 35.8877,
        "s75": 0.5004,
        "s76": 24,
        "s77": "1:19:28.750000"
    }
}
```


Figura 19. Json gerado com dados das funcionalidades da ISO 37120.

O processamento de alertas foi adicionado após a descompactação dos dados, guardando o tipo do alerta, quando houver. Após um estudo das propostas da CPFL[66], os limites de tensão possíveis de serem atingidos dentro da normalidade foram fixados em 5% do contratado. Logo, se o valor ultrapassar ou ser inferior a esta porcentagem, é considerado Sobretensão ou Subtensão respectivamente. No caso de residências com uma tensão de 220V, pode ter uma variação de 11 volts para cima ou para baixo sem causar alertas.

Segundo a NBR 5410:1997 [67], o disjuntor desliga-se automaticamente ao alcançar a sua corrente nominal. Pensando nisso, foi definido que os alertas de sobrecorrente devem acontecer somente quando a corrente estiver em 90% da corrente nominal estabelecida. Qualquer margem acima dessa porcentagem será considerado um evento de sobrecorrente. O alerta de Fuga ocorre quando os dados *rmsDiff_real* são superiores a 0.003, indicando que há corrente indevida sendo desperdiçada na residência.

4.5 DASHBOARD

A implementação do *dashboard* contou com uma série de desafios que vão ser detalhados nesta seção. A principal utilidade dessa interface gráfica é que os dados podem ser disponibilizados para usuários sem muitos detalhes técnicos, podendo ajudar a conscientizá-los sobre o uso da energia no seu dia-a-dia.

Antes do desenvolvimento das telas, foi desenvolvido o sistema de login, para que seja possível autenticar os usuários. Nesta etapa, o usuário deve informar o UUID e a senha para que seja autenticado no sistema. Após a liberação do acesso, o usuário pode acessar qualquer tela disponível para ele.

Durante a etapa de desenvolvimento do serviço, viu-se a necessidade de avaliar quais dados são realmente importantes para disponibilizar para os cidadãos. Pensando nisso, foi agendado uma reunião com o engenheiro responsável pela supervisão elétrica do Campus da UPF. Durante a conversa, foi apresentado o andamento do projeto de mes-

trado, juntamente com a dúvida de quais dados seriam realmente importantes para um usuário comum. Diante disso, chegou-se a conclusão de que os dados de tensão, corrente e consumo são os dados que mais têm importância e devem ser disponibilizados primeiro, outros dados são importantes, porém somente para fins de análise mais profunda do comportamento da rede elétrica. Nessa reunião, também obteve-se conhecimento sobre como é feita o controle do consumo elétrico do campus 1 da UPF, com apresentações dos gráficos e *dashboards* usados que contribuíram para o aprimoramento das ideais de implementação desse trabalho.

Com base nas lições aprendidas da reunião com o engenheiro de controle de energia elétrica da UPF, foi projetado o *dashboard* para disponibilização dos dados do último evento gerado. Nesta tela o usuário pode verificar a corrente e a tensão através de um calibrador. Os dados de corrente e consumo também serão disponibilizados em formato de gráfico, incrementando-se a cada novo evento captado. Haverá, por fim, um resumo geral do evento, com as informações que foram disponibilizadas em forma de gráfico ou calibrador. Caso o evento tenha um alerta, um *pop-up* é aberto na tela, notificando o usuário do ocorrido.

Buscando detalhar as informações contidas na seção 3.1.1, foi criada a tela de visualização de dados por intervalo de datas. Nesta tela, é possível selecionar o tipo de informação a ser detalhada e a data de início e fim da busca dos eventos.

A disponibilização das residências em um mapa e as informações da ISO 37120 foram agrupadas em apenas uma tela. Para alimentar o mapa com os dados, foi utilizado as rotas do serviço descritas na seção 4.4.2 para recuperar as informações do mapa e das funcionalidades da ISO 37120. Dessa forma os dados de cada residência vão ser inseridos no mapa, através de pequenos *pop-ups* com suas informações. As informações da ISO são disponibilizadas ao lado do mapa com suas respectivas identificações. Quando existe algum alerta para ser notificado, pequenas janelas vão aparecer na parte inferior da tela, com informações a respeito do alerta e da residência afetada.

4.6 INSTALAÇÃO E USO DA INTERSCITY

Durante a etapa de desenvolvimento, uma versão da InterSCity foi instalada localmente na máquina usada para o desenvolvimento do serviço de processamento, facilitando a utilização da mesma. A máquina em questão possuía sistema operacional Ubuntu com a versão 16.04, 8 GBs de memória Ram, Processador Intel Core i3 e 128 GBs de armazenamento.

A instalação da plataforma local obteve sucesso sem complicações em um primeiro momento. Porém após o início da sua utilização de forma massiva, a mesma começou a apresentar problemas na sua inicialização. Durante a etapa de desenvolvimento,

notou-se um comportamento estranho por parte da plataforma. Ao ligar a máquina e logo em seguida abrir uma página Web qualquer, a inicialização de alguns microsserviços ficava comprometido, causando o mau funcionamento da plataforma.

O problema citado acima só era corrigido, quando sistema era reinicializado por completo e, antes de qualquer ação, obrigatoriamente a plataforma deveria ser inicializada, para que a mesma pudesse configurar os seus microsserviços com sucesso.

Após uma extensa investigação e trocas de *e-mail* sem sucesso com o suporte [68], o problema foi encontrado e corrigido. A causa do problema era resultante de uma configuração que o *Docker*[69] realiza para inicializar o banco de dados não relacional(*MongoDB*) [70]. Ao inicializar o contêiner responsável pelo *data collector*, uma configuração para manter o backup dos dados locais era carregada, fazendo com que a incompatibilidade dos caminhos do sistema ocasionasse os erros. Após apagar essa configuração, a plataforma voltou a funcionar normalmente.

Vale ressaltar que, durante o desenvolvimento desta pesquisa, a USP começou a elaborar um processo para a reestruturação do código-fonte da plataforma, o que acarretaria, futuramente, em uma nova versão estável. Porém, devido a probabilidade de falhas de código que poderiam resultar em mau funcionamento, foi decidido manter a versão legada da InterSCity, por ser uma versão já validada em testes locais.

4.7 INTERSCITY NA UPF

Para validar o desenvolvimento e os objetivos do trabalho, após a estrutura básica do serviço e do *dashboard* estarem completas, foi concentrado os esforços para a estabilização da mesma versão da InterSCity em uma máquina nova hospedada no campus 1 da UPF. A máquina possui o sistema operacional Ubuntu 16.04, processador I7-3770, 8 GBs de memória e 500 Gbs de armazenamento. Vale ressaltar que devido a escassez de recursos, esta máquina também é compartilhada com outros projetos dentro do grupo de pesquisa.

Após a instalação da plataforma na nova máquina, foi necessário que a endereço IP da máquina fosse liberado para acesso externo à UPF. Para tornar isso possível, foram feitos acordos com a equipe de suporte de redes do Campus 1 da UPF para que seja possível fazer a abertura deste endereço [71].

Após a liberação do endereço IP da máquina em questão, passou a ser possível acessá-la remotamente. Por questões de segurança e configuração, somente as portas 8000 e 8888 foram abertas. A 8000 é direcionada a InterSCity, onde as requisições devem possuir a porta e o microsserviço para serem endereçados corretamente. A porta 8888 redireciona o usuário para a documentação da plataforma. Dentro desta página, é possível

encontrar todas as rotas dos microsserviços da plataforma, bem como exemplos de como acessar e filtros de busca de dados disponíveis.

Durante a etapa de testes com a plataforma, foi acordado uma troca na identificação do endereço, passando a atender pela url *http://interscity.upf.br:8000*.

A InterSCity já possui dois projetos em andamento dentro da Universidade de Passo Fundo. Um deles é o projeto descrito neste texto e o outro é um trabalho de graduação para monitoramento da movimentação e controle de frequência de alunos no campus da UPF utilizando *Beacons*. Através da leitura destes *Beacons* e a localização do *smartphone* dos alunos, é possível registrar a presença do usuário, provando que o mesmo compareceu a um local específico [72].

Encontra-se em desenvolvimento um novo trabalho de conclusão do curso de Engenharia de Computação que vai utilizar a infraestrutura desse trabalho e a InterSCity. É um projeto para monitoramento elétrico dos prédios no Campus central da UPF.

5. RESULTADOS

Durante a etapa de desenvolvimento do serviço de orquestração de chamadas para a InterSCity, foi disponibilizado o hardware de monitoramento de energia elétrica por Mella M. J. [52]. No processo de entrega, foi realizado um momento de apresentação do hardware, explicando todos os pontos do trabalho desenvolvido. Diante disso, foi feito um esforço que ajustou o serviço para receber os dados dos eventos. Durante esta etapa, foi validado com sucesso a integração do hardware com o serviço receptor. O fluxo principal de recebimento de eventos foi percorrido sem apresentar falhas de desenvolvimento.

A etapa seguinte, era validar o fluxo anterior com o envio dos dados para a plataforma InterSCity. Nesta data, ainda não estava disponível a plataforma na UPF, por esse motivo foi utilizado a plataforma local, que acabou servindo de base dos dados por todo o período de desenvolvimento. Os testes foram concluídos com sucesso, sem falhas de desenvolvimento. A partir dessa data, o hardware ficou disponível para testes dentro do campus 1 da UPF.

Após uma conversa com Mella M. J. [52] para planejamento de testes com o hardware, descobriu-se uma limitação nos componentes do hardware até então desconhecida para o projeto. O sensor responsável por ler a corrente elétrica que foram utilizados no protótipo, possuem uma limitação de leitura de corrente à 5A. Logo, as leituras acima desse valor serão incorretas e não refletem a realidade [73].

A descoberta da limitação dos sensores do hardware impediu o uso da mesma para simulação com vários equipamentos ligados ao mesmo tempo. O protótipo tinha objetivos específicos de validar os eventos, não necessitando de monitoramento da rede elétrica das residências físicas.

Durante a fase de testes foram coletados cerca de 60 eventos de *EVENT_UP* e *EVENT_DOWN*. O equipamento utilizado na rede foi uma lâmpada de 50w. Foram feitas várias abordagens de ligar e desligar a lâmpada, a fim de detectar e gerar os eventos.

Durante a fase de testes com o equipamento e a plataforma, foi publicado um artigo contendo a apresentação do projeto e as funcionalidades empregadas. O artigo foi publicado no evento SABTIC (Seminário Argentina - Brasil de Tecnologias da Informação e da Comunicação) no dia 31 de Outubro de 2019.

5.1 VALIDAÇÃO DA INTERFACE PARA VISUALIZAÇÃO DAS INFORMAÇÕES

Na tela de login do *dashboard*, o serviço verifica a autenticação do usuário, que pode utilizar todas as telas criadas, com exceção da tela de mapas. É necessária uma

conta com privilégios de Administrador para que seja possível acessar a tela, uma vez que a mesma dispõe de informações referentes a outras residências.

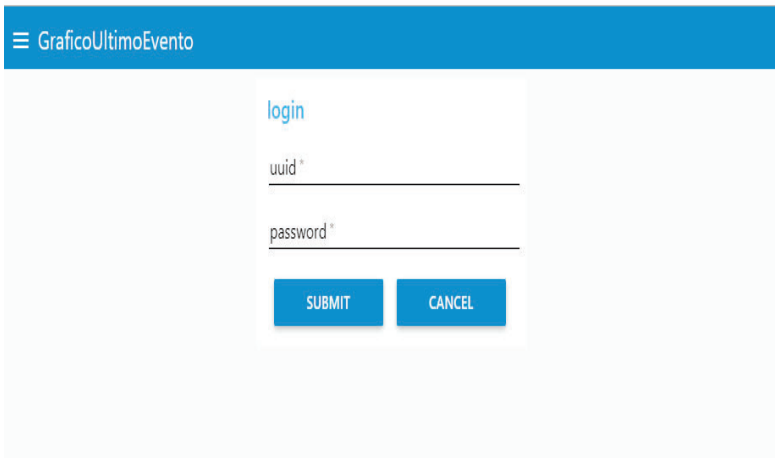

Figura 20. Tela de login para usuários comuns.

A Figura 20 mostra a tela de autenticação. Ela segue os mesmos padrões de visuais das outras telas. No menu superior esquerdo é possível visualizar as telas disponíveis, porém não possui acesso as mesmas, pois o usuário precisa estar autenticado para que o sistema libere acesso as telas.

5.1.1 Dashboards de Monitoramento

Os resultados desta seção foram baseados nas analises dos testes feitos com os *dashboards* de último evento, intervalo da datas e o mapa com as residências. Desta forma comprova-se que os *dashboards* disponibilizam as informações anteriormente projetadas, juntamente com as funcionalidades descritas na ISO 37120.

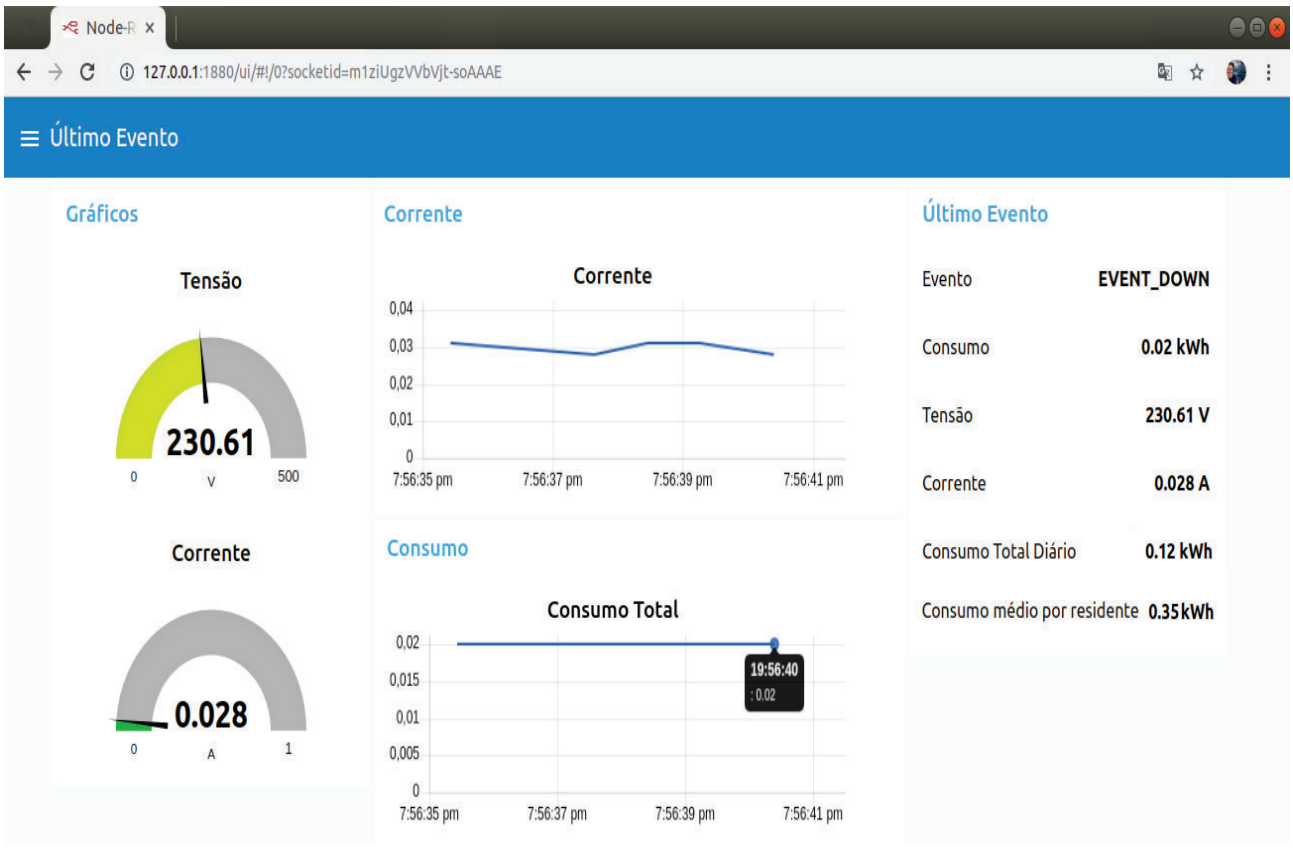

Figura 21. Tela de último evento, baseado em um evento capturado do protótipo em funcionamento.

A Figura 21 mostra o funcionamento da tela do último evento. Na parte superior esquerda foi adicionado um menu, onde todas as telas disponíveis são disponibilizadas e podem ser acessadas. Os calibradores de tensão e corrente possuem intervalos de valores para mudança de cores. No caso do calibrador de tensão, entre 0V e 230V a cor é verde, no intervalo de 230V até 250V passa a ser amarelo, indicando sobrecorrente, acima de 250V é alterado para vermelho. O mesmo acontece para o calibrador de corrente, fixado de 0A a 14A como verde e 14A a 15A como amarelo indicando sobrecorrente. Os gráficos de corrente e consumo existem para dar uma outra opção de visualização, sendo uma evolução histórica dos eventos. Para finalizar, todas as informações são listadas em formato texto, juntamente com a consumo diário total e o indicador de consumo médio da residência per capita da ISO 37120.

Na analise da Figura 21, pode-se perceber que a corrente dos eventos anteriores não variou muito, o que significa que o equipamento, por mais fraco que seja, foi desligado aproximadamente as 19:56:37 e logado novamente as 19:56:38, mantendo-se ligado por cerca de dois segundo ate ser desligado novamente. Nota-se também que o consumo não teve mudança significativa no gráfico.

Ainda sobre Figura 21, a tensão obtida não foi o suficiente para gerar um alerta no evento, portanto a cor do calibrador mudou para amarelo, para sinalizar que a tensão está próxima de se encaixar em um alerta de sobretensão. A ISO 37120 representada

pelo indicador de consumo médio da residencia per capita pode ser localizada a direita da imagem.

Figura 22. Gráfico de corrente por intervalo de tempo, baseado em dados enviados pelo protótipo.

No teste representado pela Figura 22, é resgatado o histórico da corrente dos eventos no intervalo de 29 de Agosto de 2019 a 17 de Setembro de 2019. Ao analisar o gráfico, pode-se perceber que um equipamento foi ligado e desligado varias vezes com o intervalo de alguns minutos cada sequencia. Após o fim da sequencia, a corrente manteve-se relativamente baixa, com pouca variação ate o protótipo de monitoramento elétrico ser desligado as 20:47:53 do dia 29 de Agosto de 2019.

Como as cargas de energia coletadas pelo protótipo de monitoramento são baixas, não é possível ter eventos de alerta, uma vez que para que um alerta de sobrecorrente seja gerado, é necessário que a corrente esteja acima de 90% da corrente nominal da residência.

A Figura 23 mostra o resultado do teste feito para recuperar o histórico de tensão dos eventos no intervalo de 05 de Agosto a 23 de Setembro de 2019. Em uma primeira analise é possível perceber variações significativas em curtos períodos de tempo. No período do dia 05 a 29 de Agosto, as variações foram regulares, mostrando um comportamento esperado na rede.

Figura 23. Gráfico de tensão por intervalo de tempo, baseado em dados do protótipo.

A linha vermelha no gráfico da Figura 23 demarca o limite de tensão sem que haja alertas. Após as 20 horas e 45 minutos do dia 29 de Agosto, pode-se perceber que houve um aumento notável na tensão, gerando um alerta. Após este evento, o gráfico continuou a variar bastante, gerando dois novos alertas posteriormente. O gráfico voltou a normalizar até o dia 23 de Setembro, onde gerou novos alertas.

Como demostrado nas Figuras 22 e 23, as informações de energia dos *dashboards* podem ajudar o usuário na tomada de decisão e na prevenção de problemas ligados a rede elétrica. Ao visualizar o histórico de eventos o usuário pode ter noção do comportamento da sua rede elétrica, podendo perceber problemas que não poderiam ser encontrados sem a ajuda se um técnico especializado na área.

5.1.2 Mapa de Monitoramento de Residências

Para dar uma visão geral do comportamento das residências como um todo, a tela de mapa pode ser utilizada. Todas as casas cadastradas no sistema vão ser consultadas, retornando o seu último evento captado ou o alerta que foi gerado.

A Figura 24 mostra a página do mapa em funcionamento. Após receber os dados das residências e inseri-los no mapa, os pontos referentes a cada residência aparecem junto com suas respectivas informações. Em casos de eventos normais, o ponto aparece com a letra "C", porém em casos de alerta, a letra muda para "A". Os dados das funciona-

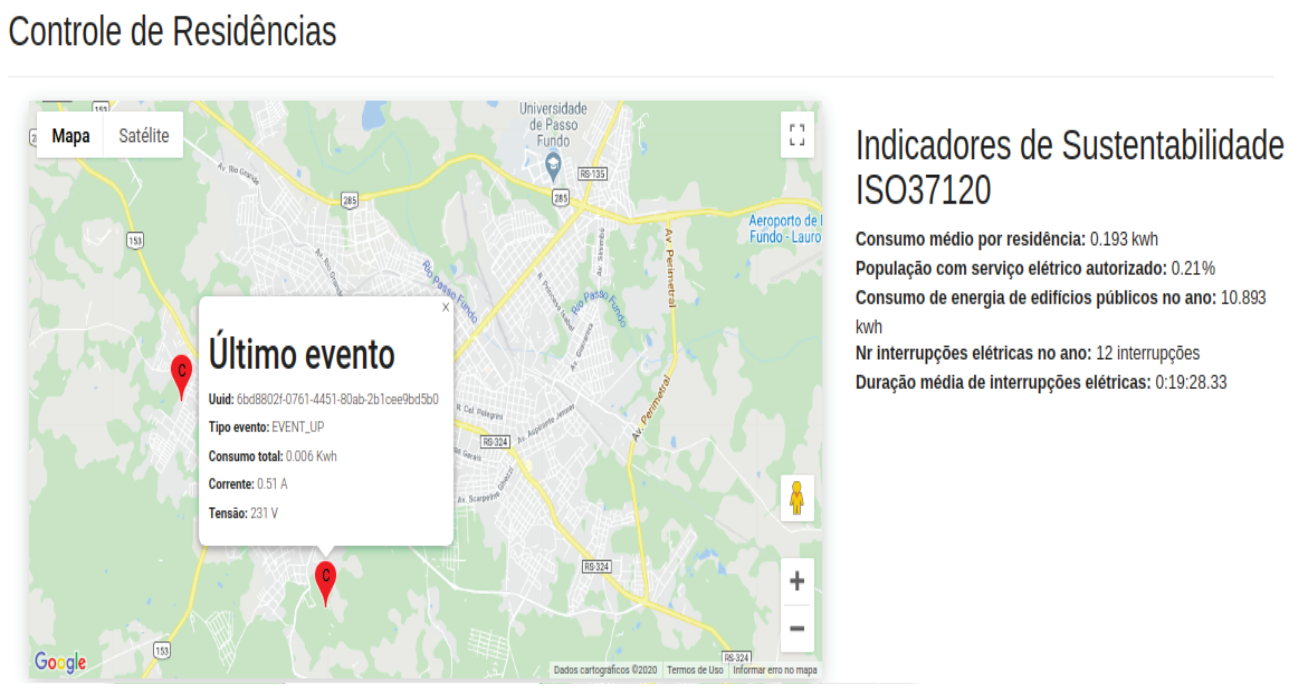

Figura 24. Tela de integração com o Google maps em um evento comum.

lidades da ISO 37120 são preenchidos na sequência. A tela possui uma atualização dos dados programada para executar a cada 2 minutos.

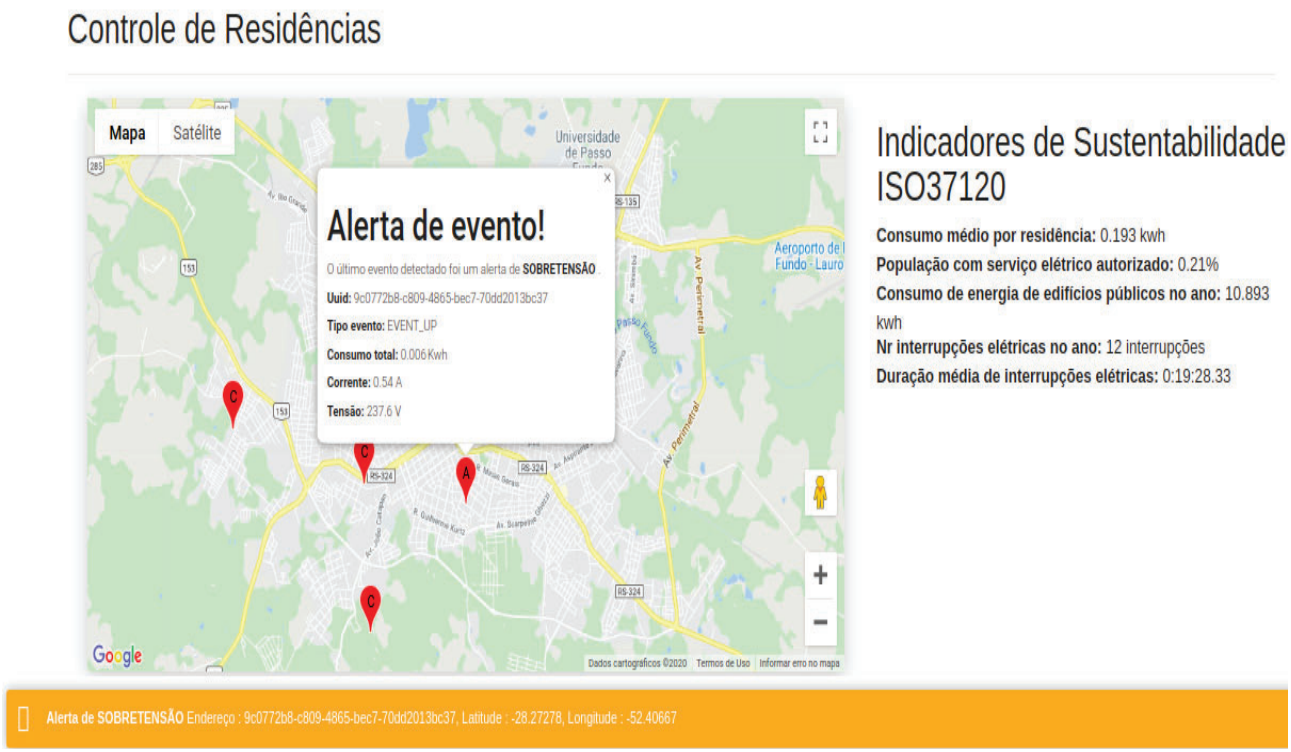

Figura 25. Tela de integração com Google maps em um evento de alerta.

A Figura 25 mostra o mapa com eventos de alerta. Quando alguma residência possui um evento de alerta, a tela do mapa atualiza o respectivo *pop-up* com o alerta, em paralelo, um aviso é inserido na tela, mostrando que houve um alerta na residência. Em cenários com poucas residências, o alerta pode ser vantajoso para evidenciar problemas.

5.2 SIMULADOR DE RESIDÊNCIAS

Como o objetivo do trabalho é monitorar energia elétrica de residências em ambientes de cidades inteligentes, o teste com apenas um hardware físico não entregaria o resultado esperado para um ambiente urbano. Para contornar este obstáculo, duas soluções eram possíveis: Uso do *InterSCSimulator* [39] ou a criação de um *script* de teste para simular várias casas.

Por ser um software naturalmente citado nos repositórios do InterSCity como uma solução para simuladores de ambientes, o *InterSCSimulator* seria analisado com o objetivo de usá-lo como um simulador de residências. Porém, ao iniciar as análises, notou-se que o programa é focado em geração de dados de tráfego urbano, necessitando de uma série de configurações relacionadas a tráfego.

Por se tratar de simulador de tráfego urbano, o seu uso ficou bastante limitado por possuir um objetivo bem específico. Logo, não seria vantajoso para o projeto, visto que as alterações a serem feitas seriam trabalhosas demais: estudo completo do código-fonte do programa, criar um fluxo alternativo para geração de dados fictícios, etc.

Com o uso do *InterSCSimulator* descartado, optou-se por criar um *script* para simular várias residências em uma cidade. Os eventos gerados foram baseados nos eventos do hardware utilizados anteriormente. É importante salientar que estes eventos eram gerados com uma carga baixa, ou seja, somente um equipamento ligado a tomada do hardware, geralmente uma lâmpada ou algo similar.

A simulação de eventos por *script* também acabou tendo que ser adotada como *case* para testes de performance, tanto no serviço quanto no InterSCity, validando se a plataforma suporta a quantidade de dados enviados e se o serviço consegue suprir a demanda de processamento dos eventos.

O primeiro passo da execução do serviço foi gerar casas aleatórias, com a latitude e longitude iniciando no centro de Passo Fundo e variando aleatoriamente. Todos os dados para criação de residências eram gerados aleatoriamente pelo *script*. Caso não necessitasse criar novas residências, era buscado todas as existentes no banco de dados, utilizando-as para gerar os dados.

Após a seleção dos eventos bases para criação dos dados fictícios, definiu-se as regras de variação dos eventos, isso significa que os dados principais descritos na seção 3.1.1 poderiam variar em uma porcentagem estipulada para cada um. A primeira regra foi

a definição do tipo do evento. Se for selecionado o evento do tipo *EVENT_UP*, os dados variariam para um valor acima do padrão, caso contrário, variariam para baixo.

O problema desses eventos gerados centralizou-se na geração de eventos de alerta. Como os eventos gerados tinham como padrão uma carga relativamente baixa composta por apenas um equipamento, era impossível de gerar um evento de sobrecorrente, uma vez que os mesmos não atingem uma corrente superior a 90% do suportado. Eventos de Sobretensão Subtensão e Fuga são mais fáceis de gerar, uma vez que a porcentagem de variação destes dados podem chegar a 10%.

Para controlar a geração de alertas no *script*, foi inicialmente utilizado a probabilidade de 5% para criar um evento de alerta. Desta forma, os testes não devem apresentar uma quantidade expressiva de alertas, gerando uma interpretação de uma residencia com poucos problemas elétricos.

Durante a fase de testes com o *script*, foram gerados cerca de 6620 eventos, distribuídos em 278 casas diferentes. Os eventos foram gerados no mês de janeiro, variando conforme a execução do *script*. As sequências de testes eram compostas pela geração de dois eventos para cada residência.

Durante a execução de uma sequência de teste no dia 28 de janeiro de 2019, observou-se a geração de eventos e seus alertas. Foram gerados 277 eventos, sendo que era esperado que cerca de 13 eventos fossem gerados como alertas. Após a finalização do teste, 11 eventos de alerta foram criados, totalizando 3.97% de alertas.

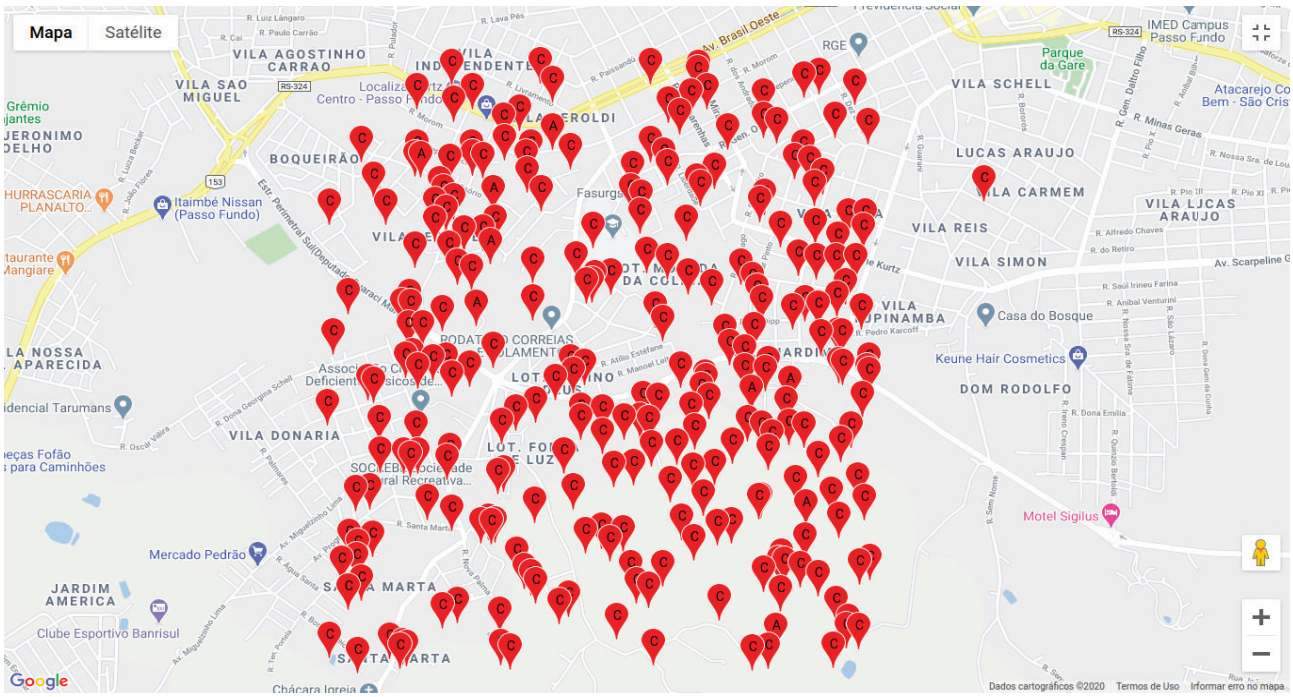

Figura 26. Mapa com as residências e seus últimos eventos.

A Figura 26 mostra o mapa de residências após a finalização da execução dos testes do dia 28 de janeiro de 2019. Após análise, é possível confirmar que poucas residências tiveram eventos de alerta como sendo seus últimos eventos.

Durante a execução dos testes e a visualização do mapa, notou-se um atraso no tempo de processamento no serviço. Cada residência era tratada individualmente no processo, buscando os eventos na plataforma e processando-os. Após, o processo se repetia para gerar o cálculo das funcionalidades da ISO 37120. Todo o processo completo teve um tempo de resposta de 1 minuto e 20 segundos.

Após a análise de código, ficou evidente o problema de performance, com cada residência levando mais de um minuto para processar. Para tentar corrigir o problema, foi explorado mais tecnicamente as rotas disponíveis na InterSCity para achar uma forma de pesquisar os eventos de todos os UUIDs de uma única vez. Durante a análise, constatou-se que a plataforma possui a possibilidade de pesquisa por vários UUIDs. Após a correção deste problema, o tempo de resposta da requisição diminuiu para 31 segundos.

Apos a análise dos resultados, ficou entendível que as normas especificadas na ISO 37120 foram atendidas e disponibilizadas de maneira intuitiva nos *dashboards* implementados. Espera-se que com essas informações, os usuários possam tomar conhecimento sobre a situação da sua rede elétrica, auxiliando nas tomadas de decisão.

6. CONSIDERAÇÕES FINAIS E TRABALHOS FUTUROS

O conceito de cidades inteligentes está se consolidando cada vez mais dentro dos ambientes urbanos. A cada dia, novos processos surgem para suprir demandas por segurança e qualidade de vida para os cidadãos. Estes processos necessitam de uma grande quantidade de dados para serem elaborados e testados. O presente trabalho teve como objetivo apresentar uma solução para a disponibilização de informações de energia elétrica residencial em um ambiente de cidades inteligentes através de um protótipo de monitoramento de energia elétrica e da plataforma InterSCity.

Disponibilizar informações a respeito de um determinado tema pode ajudar o usuário final na tomada de decisão. No contexto de cidades inteligentes, o auxílio na tomada de decisão através de sistemas de ciência de dados é essencial. Porém, esses sistemas só serão eficientes se tiverem uma boa base de dados para pesquisa. O intuito deste trabalho, no âmbito de uma cidade inteligente, foi disponibilizar uma forma de armazenamento e disponibilização de dados de energia através da plataforma InterSCity.

O armazenamento de dados de energia elétrica em residências é um caminho ainda pouco explorado. A sua exploração pode trazer grandes benefícios para os cidadãos, auxiliando na conscientização do consumo e na descoberta de problemas relacionado a rede elétrica.

O presente trabalho mostrou que a união de uma plataforma para cidades inteligentes, e um projeto de monitoramento de energia elétrica integrados em uma cidade, geram informações que podem ser relevantes para os cidadãos, podendo conscientizá-los sobre a atual situação da sua rede elétrica residencial.

A plataforma InterSCity cumpriu o seu papel de ser uma estrutura robusta, oferecendo suporte para o desenvolvimento do projeto. Neste momento, a plataforma oferece suporte para duas aplicações desenvolvidas no campus 1 da UPF. Além disso, a plataforma está disponível para que novos projetos possam utilizá-lo.

Os objetivos propostos foram atingidos neste trabalho. A integração entre o hardware, serviço e InterSCity obteve êxito. O serviço apresentou a performance esperada ao orquestrar várias chamadas do *script* para criação de novas residências e o envio dos eventos das mesmas. Na mesma linha seguiu a InterSCity, entregando exatamente o desempenho esperado, tanto na busca quanto no cadastro de novos eventos.

A implantação do hardware de monitoramento de eventos em uma residência, poderia contribuir para a geração de dados em tempo real, para que seja possível recalibrar o *script* para a geração de eventos mais equivalentes a uma residência com inúmeros equipamentos ligados a rede elétrica. Nos trabalhos futuros, será necessário alterar os sensores

que estão limitados a leitura de corrente de 5A, para um que consiga suportar a demanda de uma residência sem gerar dados inconsistentes.

Para contemplar o indicador de energias renováveis presente na ISO 37120, o protótipo de monitoramento de Mella M.J. deve ser alterado de forma que consiga detectar cargas elétricas vindas de fontes renováveis [52]. Desta forma, todos os indicadores da ISO serão atendidos neste trabalho.

Para uma maior imersão do usuário, o *dashboard* deve ser reprojetado, dessa vez em uma ferramenta específica que permita a personalização de gráficos e tabelas e outras ferramentas. Esta mudança pode fazer com que o usuário tenha mais opções para filtrar ou otimizar sua busca pelos dados de energia elétrica. Além disso, faz-se necessário a refatoração dos processamentos dos dados para o mapa de residências com o objetivo de reduzir ainda mais o tempo de resposta descrito nos resultados obtidos.

O desenvolvimento deste trabalho abriu muitas portas para novos projetos na área de monitoramento de energia elétrica. Novos projetos de monitoradores de energia podem ser usados para geração de dados, explorando outras áreas mais específicas que podem ajudar a prevenir problemas relacionados a rede elétrica.

As interfaces de visualização de dados criadas para este trabalho contribuirão para um projeto de graduação no curso de Engenharia da Computação. Será coletado e analisado dados de energia dos prédios do Campus da UPF, disponibilizando-os nos *dashboards* deste trabalho, além da integração com o InterSCity para a manutenção dos dados.

Novas interfaces podem ser criadas com a exploração dos dados armazenados no InterSCity. Telas com novas funcionalidades, como por exemplo, uma previsão do valor a ser pago pela energia elétrica utilizada podem ajudar o usuário a conscientizar-se sobre o consumo.

Com a plataforma estabilizada e testada, novos projetos podem utilizar das ferramentas disponíveis, criando um ambiente de cidades inteligentes com a disponibilização de várias bases de dados para todos os usuários da rede.

REFERÊNCIAS BIBLIOGRÁFICAS

[1] HARRISON, C.; DONNELLY, I. A. A theory of smart cities. Proceedings of the 55th Annual Meeting of the ISSS - 2011, Hull, UK, v. 55, n. 1, Sep. 2011.

[2] Celino, I.; Kotoulas, S. Smart cities [guest editors' introduction]. IEEE Internet Computing, v. 17, n. 6, p. 8–11, Nov 2013. ISSN 1089-7801.

[3] LEA, R. Smart Cities: An Overview of the Technology Trends Driving Smart Cities. 2017.

[4] GAUR, A. et al. Smart city architecture and its applications based on iot. Procedia Computer Science, v. 52, p. 1089–1094, 12 2015.

[5] SMART Cities Council. Disponível em: <https://smartcitiescouncil.com/article/resources>. Acesso em: 2018-04-12.

[6] BUCKMAN, A.; MAYFIELD, M.; BECK, S. B. What is a smart building? Smart and Sustainable Built Environment, Emerald, v. 3, n. 2, p. 92–109, sep 2014. Disponível em: <https://doi.org/10.1108/sasbe-01-2014-0003>.

[7] JINDARAT, S.; WUTTIDITTACHOTTI, P. Smart farm monitoring using raspberry pi and arduino. 2015 International Conference on Computer, Communications, and Control Technology (I4CT), p. 284–288, 2015.

[8] GUNGOR, V. C. et al. Smart grid technologies: Communication technologies and standards. IEEE Transactions on Industrial Informatics, v. 7, n. 4, p. 529–539, Nov 2011. ISSN 1551-3203.

[9] PATTI, E. et al. Distributed Software Infrastructure for General Purpose Services in Smart Grid. IEEE Transactions on Smart Grid, v. 7, n. 2, p. 1156–1163, mar 2016. ISSN 1949-3053.

[10] BAWANY, N.; SHAMSI, J. Smart city architecture: Vision and challenges. v. 6, 11 2015.

[11] SOUZA, A. F. de et al. A data integration approach for smart cities: The case of natal. 2017 International Smart Cities Conference (ISC2), p. 1–6, 2017.

[12] PETROLO, R.; LOSCRÍ, V.; MITTON, N. Towards a smart city based on cloud of things. In: Proceedings of the 2014 ACM International Workshop on Wireless and Mobile Technologies for Smart Cities. New York, NY, USA: Association for Computing Machinery, 2014. (WiMobCity '14), p. 61–66. ISBN 9781450330367.

[13] REDA, F.; KARJALAINEN, S.; TUOMISTO, M. Combined use of nonintrusive monitoring techniques and energy recipes to reduce energy hungry behaviours. 2017 International Conference on Engineering, Technology and Innovation (ICE/ITMC), p. 290–296, 2017.

[14] JINMENG, Z.; XIAOMING, W. Key technologies of medical monitoring system of smart home. 2011 4th International Congress on Image and Signal Processing, p. 190–193, 10 2011.

[15] GOVINDRAJ, V.; SATHIYANARAYANAN, M.; ABUBAKAR, B. Customary homes to smart homes using internet of things (iot) and mobile application. 2017 International Conference On Smart Technologies For Smart Nation (SmartTechCon), p. 1059–1063, 2017.

[16] FISCHER, C. Feedback on household electricity consumption: a tool for saving energy? Energy Efficiency, Springer Science and Business Media LLC, v. 1, n. 1, p. 79–104, fev. 2008.

[17] SILVA, E. R. P. Métodos para Revisão e Mapeamento Sistemático da Literatura (Methods for Systematic Literature Reviews and Systematic Mapping Studies). 2009. Artigo (Engenharia de Produção), UFRJ (Universidade Federal do Rio de Janeiro), Rio de Janeiro, Brasil.

[18] GARCIA, V. C.; ALVARO, A.; GAMA, K. Smart Cities Architectures - A Systematic Review. In: Proceedings of the 15th International Conference on Enterprise Information Systems. Angers, France: SciTePress - Science and and Technology Publications, 2013. p. 410–417. ISBN 978-989-8565-59-4.

[19] SANTANA, E. F. Z. et al. Software Platforms for Smart Cities. ACM Computing Surveys, v. 50, n. 6, p. 1–37, nov 2017. ISSN 03600300.

[20] M. Del Esposte, A. et al. InterSCity: A Scalable Microservice-based Open Source Platform for Smart Cities. In: Proceedings of the 6th International Conference on Smart Cities and Green ICT Systems. Porto, Portugal: SCITEPRESS - Science and Technology Publications, 2017. p. 35–46. ISBN 978-989-758-241-7.

[21] BATISTA, D. M. et al. Interscity: Addressing future internet research challenges for smart cities. In: 2016 7th International Conference on the Network of the Future (NOF). Buzios, Brasil: IEEE, 2016. p. 1–6.

[22] PATTI, E. et al. District Information Modeling and Energy Management. IT Professional, v. 17, n. 6, p. 28–34, nov 2015. ISSN 1520-9202.

[23] BRUNDU, F. G. et al. IoT software infrastructure for Energy Management and Simulation in Smart Cities. v. 3203, n. c, p. 832–840, 2016.

[24] BONINO, D. et al. ALMANAC: Internet of Things for Smart Cities. In: 2015 3rd International Conference on Future Internet of Things and Cloud. Roma, Itália: IEEE, 2015. p. 309–316. ISBN 978-1-4673-8103-1.

[25] BONINO, D.; De Russis, L. Complex Event Processing for City Officers: A Filter and Pipe Visual Approach. IEEE Internet of Things Journal, v. 5, n. 2, p. 775–783, apr 2018. ISSN 2327-4662.

[26] SANCHEZ, L. et al. Smartsantander: The meeting point between future internet research and experimentation and the smart cities. In: 2011 Future Network Mobile Summit. Varsóvia, Polônia: IEEE, 2011. p. 1–8.

[27] SANCHEZ, L. et al. SmartSantander: IoT experimentation over a smart city testbed. Computer Networks, Elsevier B.V., v. 61, p. 217–238, mar 2014. ISSN 13891286.

[28] VILLANUEVA, F. J. et al. Civitas: The Smart City Middleware, from Sensors to Big Data. In: 2013 Seventh International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing. Taichung, Taiwan: IEEE, 2013. p. 445–450. ISBN 978-0-7695-4974-3. ISSN 0213-4691.

[29] APOLINARSKI, W.; IQBAL, U.; PARREIRA, J. X. The GAMBAS middleware and SDK for smart city applications. In: 2014 IEEE International Conference on Pervasive Computing and Communication Workshops (PERCOM WORKSHOPS). Budapest, Hungria: IEEE, 2014. p. 117–122. ISBN 978-1-4799-2736-4. ISSN 2474-2503.

[30] MADUKWE, K. J.; EZIKA, I. J. F.; ILOANUSI, O. N. Leveraging edge analysis for Internet of Things based healthcare solutions. In: 2017 IEEE 3rd International Conference on Electro-Technology for National Development (NIGERCON). Owerri, Nigéria: IEEE, 2017. p. 720–725. ISBN 978-1-5090-6422-9.

[31] FERREIRA, D. et al. Towards smart agriculture using FIWARE enablers. In: 2017 International Conference on Engineering, Technology and Innovation (ICE/ITMC). Funchal, Portugal: IEEE, 2017. p. 1544–1551. ISBN 978-1-5386-0774-9.

[32] STEINMETZ, C. et al. Ontology-driven IoT code generation for FIWARE. In: 2017 IEEE 15th International Conference on Industrial Informatics (INDIN). Emden, Alemanha: IEEE, 2017. p. 38–43. ISBN 978-1-5386-0837-1.

[33] ATZORI, L.; IERA, A.; MORABITO, G. The Internet of Things: A survey. Computer Networks, Elsevier B.V., v. 54, n. 15, p. 2787–2805, 2010. ISSN 13891286.

[34] AL-HADER, M. et al. Smart City Components Architicture. In: 2009 International Conference on Computational Intelligence, Modelling and Simulation. Brno, República Tcheca: IEEE, 2009. p. 93–97. ISBN 978-1-4244-5200-2. ISSN 2166-8523.

[35] HANDTE, M. et al. The base plug-in architecture - composable communication support for pervasive systems. 7th ACM International Conference on Pervasive Services, 07 2010.

[36] TECHNOLOGIES, K. Kaa Project. Disponível em: <https://www.kaaproject.org>.

[37] FIWARE. Fiware Project. 2018. Disponível em: <https://www.fiware.org/>. Acesso em: 27 Set. 2018.

[38] BRIZZI, P. et al. Towards an ontology driven approach for systems interoperability and energy management in the smart city. In: 2016 International Multidisciplinary Conference on Computer and Energy Science (SpliTech). Split, Croácia: IEEE, 2016. p. 1–7.

[39] INTERSCITY. InterSCity Platform. 2017. http://interscity.org/. Acesso em 28 Jul. 2019.

[40] EUGSTER, P. T. et al. The many faces of publish/subscribe. ACM Comput. Surv., ACM, New York, NY, USA, v. 35, n. 2, p. 114–131, jun. 2003. ISSN 0360-0300.

[41] GUTIÉRREZ, V. et al. SmartSantander: Internet of Things Research and Innovation through Citizen Participation. In: GALIS, A.; GAVRAS, A. (Ed.). Interoperability research for networked enterprises applications and software. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, (Lecture Notes in Computer Science, 5600 Blatt 3). p. 173–186. ISBN 978-3-642-38081-5.

[42] ALLIANCE, O. M. NGSI Context Management. 2010. Disponível em: <http://www.openmobilealliance.org/release/NGSI/V1_0-20120529-A/OMA-TS-NGSI_Context_Management-V1_0-20120529-A.pdf>. Acesso em: 23 Set. 2018.

[43] BARRETO, L. et al. Identity management in iot clouds: A fiware case of study. In: 2015 IEEE Conference on Communications and Network Security (CNS). Florence, Itália: IEEE, 2015. p. 680–684.

[44] MATTERN, F.; FLOERKEMEIER, C. From active data management to event-based systems and more. In: SACHS, K.; PETROV, I.; GUERRERO, P. (Ed.). Berlin, Heidelberg: Springer-Verlag, 2010. cap. From the Internet of Computers to the Internet of Things, p. 242–259. ISBN 3-642-17225-3, 978-3-642-17225-0.

[45] NALBANDIAN, S. A survey on internet of things: Applications and challenges. In: 2015 International Congress on Technology, Communication and Knowledge (ICTCK). Mashhad, Irã: IEEE, 2015. p. 165–169.

[46] SPíNOLA, R. O.; TRAVASSOS, G. H. Towards a framework to characterize ubiquitous software projects. Information and Software Technology, v. 54, n. 7, p. 759 – 785, 2012. ISSN 0950-5849.

[47] GOPALKRISHNAN, V. et al. Big data, big business: Bridging the gap. In: Proceedings of the 1st International Workshop on Big Data, Streams and Heterogeneous Source Mining: Algorithms, Systems, Programming Models and Applications. New York, NY, USA: ACM, 2012. (BigMine '12), p. 7–11. ISBN 978-1-4503-1547-0.

[48] MAYER-SCHNBERGER, V. Big Data: A Revolution That Will Transform How We Live, Work and Think. UK: John Murray Publishers, 2013. ISBN 1848547927, 9781848547926.

[49] BIGMINE '12: Proceedings of the 1st International Workshop on Big Data, Streams and Heterogeneous Source Mining: Algorithms, Systems, Programming Models and Applications. New York, NY, USA: ACM, 2012. ISBN 978-1-4503-1547-0.

[50] FILIPPONI, L. et al. Smart city: An event driven architecture for monitoring public spaces with heterogeneous sensors. In: 2010 Fourth International Conference on Sensor Technologies and Applications. Venice, Itália: IEEE, 2010. p. 281–286.

[51] ISO. ISO 37120. 2018. https://www.iso.org/standard/68498.html. Acesso em 27 Jul. 2019.

[52] MELLA, M. J. Medidor de energia elétrica por evento. Dissertação (Mestrado) — PPGCA - Universidade de Passo Fundo, Av. Brasil Leste, 285 - São José, Passo Fundo - RS, 99052-900, 2019.

[53] SANTOS, J. C. Atualização de hardware e firmware do Protegemed. Dissertação (Mestrado) — PPGCA - Universidade de Passo Fundo, Av. Brasil Leste, 285 - São José, Passo Fundo - RS, 99052-900, 2017.

[54] CCS, C. C. S. Integrated Development Environment (IDE) for TM4x ARM MCUs. 2015. http://www.ti.com/tool/ccstudio-tm4x. Acesso em 01 Ago. 2019.

[55] AZZINI, H. A. D. Avaliação de técnicas para monitoramento não intrusivo de cargas residenciais com fins de auditoria energética. Dissertação (Mestrado) — Dissertação Mestrado. PPGEE, UFES, Vitória, 2012.

[56] RUBY. Ruby 2.7.0. 1995. Disponível em: <https://www.ruby-lang.org/>. Acesso em: 28 Jan. 2019.

[57] FOUNDATION, P. S. Python. 2019. https://www.python.org/. Acesso em: 28 Jan. 2020.

[58] PROJECTS, T. P. Flask. 2019. https://palletsprojects.com/p/flask/. Acesso em: 28 Jan. 2020.

[59] SOUZA, F. V. M. UMA ARQUITETURA LPWAN DE CUSTO ACESSÍVEL PARA CIDADES INTELIGENTES. Dissertação (Mestrado) — PPGCA - Universidade de Passo Fundo, Av. Brasil Leste, 285 - São José, Passo Fundo - RS, 99052-900, 2019.

[60] FOUNDATION, O. Node-RED: Low-code programming for event-driven applications. 2013. Disponível em: <https://nodered.org/>. Acesso em: 28 Jan. 2020.

[61] FOUNDATION, O. node-red-dashboard 2.19.3. 2019. Disponível em: <https://flows.nodered.org/node/node-red-dashboard>. Acesso em: 28 Jan. 2019.

[62] FOUNDATION, O. node-red-contrib-maps 0.2.2. 2013. https://flows.nodered.org/node/node-red-contrib-maps. Acesso em: 28 Jan. 2020.

[63] PLATFORM, G. C. Google Maps Platform. 2019. Disponível em: <https://cloud.google.com/maps-platform/maps/>. Acesso em: 28 Jan. 2020.

[64] CORPORATION, O. Mysql. 2019. Disponível em: <https://www.mysql.com/>. Acesso em: 28 Jan. 2020.

[65] FOUNDATION, T. jQuery. Ajax. 2019. Disponível em: <https://api.jquery.com/category/ajax/>. Acesso em: 28 Jan. 2019.

[66] ELETRICA, A. N. de E. ANEEL. 2001. Disponível em: <http://www2.aneel.gov.br/aplicacoes/Audiencia_Publica/audiencia_proton/2001/ap004/AP004_2001_PropostaCPFL.pdf>. Acesso em: 28 Jan. 2019.

[67] TECNICAS, A. B. de N. Projeto NBR 5410:1997. 1997. Disponível em: <https://www.prolazer.com.br/download/9a1158154dfa42caddbd0694a4e9bdc8>. Acesso em: 28 Jan. 2019.

[68] GMAIL. Email de erro na inicialização do InterSCity. 2019. Disponível em: <https://mail.google.com/mail/u/0/?tab=rm\&ogbl\#search/hamario\%40ime.usp.br/QgrcJHsbhPBhTWrwxxGFFvBdMwPcNfxPJLL>. Acesso em: 28 Jan. 2020.

[69] DOCKER, I. Docker. 2013. Disponível em: <https://www.docker.com/>. Acesso em: 29 Jan. 2020.

[70] MONGODB, I. Mongo DB. 2007. Disponível em: <https://www.mongodb.com/>. Acesso em: 29 Jan. 2020.

[71] GMAIL. Acesso a servidor e Upgrade de Memória. 2019. Disponível em: <https://mail.google.com/mail/u/0/?tab=rm\&ogbl\#starred/FMfcgxwDrlZGwbKJkCRBqtRpCbzpFrxF>. Acesso em: 28 Jan. 2020.

[72] ZANELA, G. E.; REBONATTO, T. M. Smart Campus utilizando beacons: Uma aplicação para rastreamento e monitoramento de concentração de pessoas. 2019. Artigo (Bacharel em Ciência da Computação), UPF (Universidade de Passo Fundo), Passo Fundo, Brasil.

[73] GMAIL. Limitação do protótipo. 2019. Disponível em: <https://mail.google.com/mail/u/0/?tab=rm\#starred/KtbxLwgdfHjwqjDJHwhlsdvjsHtMRmjlDV>. Acesso em: 28 Jan. 2020.

UPF
UNIVERSIDADE
DE PASSO FUNDO

UPF Campus I - BR 285, São José
Passo Fundo - RS - CEP: 99052-900
(54) 3316 7000 - www.upf.br