UNIVERSIDADE FEDERAL DA PARAÍBA
CENTRO DE INFORMÁTICA

GABRIEL GUTIERREZ PEREIRA SOARES

PROJETO UBIQUE:

Rede de Sensores sem Fios para Edificações Inteligentes

João Pessoa
2018

GABRIEL GUTIERREZ PEREIRA SOARES

PROJETO UBIQUE:

Rede de Sensores sem Fios para Edificações Inteligentes

Monografia apresentada no curso de Engenharia de Computação do Centro de Informática da Universidade Federal da Paraíba, como requisito para obtenção do grau de Bacharel em Engenharia de Computação

Orientador: PhD Eudisley Gomes dos Anjos

João Pessoa

Junho de 2018

Catalogação na publicação
Seção de Catalogação e Classificação

S676p Soares, Gabriel Gutierrez Pereira.
Projeto Ubique: Rede de Sensores sem Fios para Edificações Inteligentes / Gabriel Gutierrez Pereira Soares. - João Pessoa, 2018.
57 f. : il.

Orientação: Eudisley Gomes dos Anjos.
Monografia (Graduação) - UFPB/CI.

1. edificações inteligentes. 2. internet das coisas. 3. redes de sensores sem fios. 4. sistema de controle de acesso. I. Anjos, Eudisley Gomes dos. II. Título.

UFPB/BC

CENTRO DE INFORMÁTICA
UNIVERSIDADE FEDERAL DA PARAÍBA

Trabalho de Conclusão de Curso de Engenharia de Computação intitulado **Projeto Ubique: Rede de Sensores sem Fios para Edificações Inteligentes** de autoria de **Gabriel Gutierrez Pereira Soares**, aprovada pela banca examinadora constituída pelos seguintes professores:

Prof. Dr. Eudisley Gomes dos Anjos
Universidade Federal da Paraíba

Prof. Dr. Fernando Menezes Matos
Universidade Federal da Paraíba

Profa. Dra. Camila Mara Vital Barros
Universidade Federal da Paraíba

Coordenadora do Curso de Engenharia de Computação
Thaís Gaudêncio do Rêgo
CI/UFPB

João Pessoa, 14 de junho de 2018

Centro de Informática, Universidade Federal da Paraíba
Rua dos Escoteiros, Mangabeira VII, João Pessoa, Paraíba, Brasil CEP: 58058-600
Fone: +55 (83) 3216 7093 / Fax: +55 (83) 3216 7117

AGRADECIMENTOS

Agradeço primeiramente à minha família: à minha mãe, Djanira, por todo o carinho, cuidado e atenção que deu e tem me dado desde antes do meu nascimento; ao meu pai, Manoel, por ser um referencial de caráter, honra e sabedoria na minha vida; e à minha irmã, Nicole, por todos os momentos de alegria que me proporciona.

Agradeço aos meus melhores amigos: Adriano, Cláudio, D'Angelles, Fernando, Luana e Gabriel e Tiago Daltro, os quais tive a honra de ter ao meu lado há mais de uma década, e que eu espero poder um dia retribuir toda atenção e companheirismo que me deram.

Agradeço ao Laboratório de Sistemas Digitais (LASID): por ter sido para mim uma segunda casa na UFPB e no Centro de Informática, me acolhendo durante toda a minha graduação; e por ter provido toda a infraestrutura e equipamentos (especialmente a fresadora e a impressora 3D) sem os quais não seria possível alcançar o nível de qualidade deste trabalho. Serei eternamente grato aos professores José Antônio e Hamilton Soares por terem depositado confiança em mim para participar do laboratório.

Agradeço especialmente aos professores: Eudisley Gomes, por acreditar no meu potencial, promover a confiança no meu trabalho e não ter me deixado desistir; Mardson Amorim, por ter sempre me motivado a ir cada vez mais longe; e Thaís Gaudencio, pela sabedoria e capacidade de ser rígida e ao mesmo tempo mãe de todos os alunos do curso de Engenharia de Computação.

Agradeço aos meus colegas de trabalho: Jônatas, Hélder e Daiana, e à minha chefe Diana do Instituto Federal da Paraíba, por todo o apoio, compreensão e companheirismo durante esse tempo que passei dividido entre o trabalho e a graduação.

Agradeço à Universidade Federal da Paraíba e a todos os docentes que contribuíram para a minha formação acadêmica e profissional; e aos meus colegas e amigos que acompanhei durante minha graduação.

"Se eu vi mais longe, foi por estar sobre ombros de gigantes."

(Isaac Newton)

RESUMO

Os edifícios comerciais e residenciais estão se tornando cada vez mais modernos e eficientes por meio da adoção de tecnologias inteligentes. Ao mesmo tempo, a miniaturização e o barateamento da tecnologia permitiram o surgimento de uma nova classe de dispositivos, os objetos inteligentes, que juntos formam a Internet das Coisas. Esta área tem crescido nos últimos anos, acelerando e aprofundando o uso de tecnologias nos edifícios inteligentes. Nesse contexto, este trabalho apresenta o desenvolvimento de uma rede de sensores sem fios, intitulada Projeto Ubique, com o objetivo de prover inteligência para edifícios. O Centro de Informática da UFPB foi utilizado como estudo de caso. Foi desenvolvido um sistema de controle de acesso completo (hardware e software) para validar a rede de sensores concebida. O hardware foi baseado no microcontrolador Espressif ESP8266EX e na tecnologia RFID de 13,56 MHz. O software foi desenvolvido com o auxílio dos *frameworks* Arduino (sistema embarcado) e Laravel (servidor web). Foram identificadas algumas questões de segurança que, por apresentarem peculiaridades no contexto da Internet das Coisas, vão além do escopo deste trabalho. O desenvolvimento deste projeto permitiu a validação dos conhecimentos adquiridos durante todo o curso de Engenharia de Computação, especialmente nas áreas de Engenharia de Software, Sistemas Embarcados e Redes de Sensores sem Fios.

Palavras-chave: edifícios inteligentes. internet das coisas. redes de sensores sem fios. sistema de controle de acesso.

ABSTRACT

Commercial and residential buildings are becoming more modern and efficient as a consequence of the use of smart technologies. At the same time, the decrease of production costs and miniaturization of new technologies allowed the rise of a new device group called smart objects that together make the Internet of Things. This area has grown in the last years, accelerating and spreading the usage of technologies in smart buildings. In this context, this work presents the development of a wireless sensor network called Ubique Project that aims to provide buildings with intelligence. The Centro de Informática at Universidade Federal da Paraíba was used as a case study. A complete access control system (hardware and software) was developed to validate the wireless sensor network. The hardware was based on Espressif ESP8266EX microcontroller and 13.56 MHz RFID technology. The software was developed with the aid of Arduino (embedded system) and Laravel (web server) frameworks. Some security issues were raised, but they were not addressed in this work because they are inherent to the Internet of Things area and are still being evaluated. The development of this project allowed the validation of knowledge acquired during all Computer Engineering course, specially in areas such as Software Engineering, Embedded Systems and Wireless Sensor Networks.

Keywords: smart buildings. internet of things. wireless sensor network. access control system.

LISTA DE ILUSTRAÇÕES

Figura 1 – Características de um edifício inteligente. 16
Figura 2 – Integração entre diferentes sistemas prediais. 16
Figura 3 – Arquitetura da rede de sensores sem fios do Projeto Ubique. 20
Figura 4 – Módulo de desenvolvimento WEMOS D1 Mini. 22
Figura 5 – Módulo de desenvolvimento RFID-RC522. 23
Figura 6 – Arquitetura de hardware do terminal de controle de acesso. 24
Figura 7 – Parte do circuito elétrico do terminal de controle de acesso. 25
Figura 8 – Frente e verso de uma matriz de contatos. 26
Figura 9 – Fresadora LPKF ProtoMat S42. 27
Figura 10 – Parte de uma placa de circuito impresso. 27
Figura 11 – Paquímetro digital. 28
Figura 12 – Impressora 3D Cliever CL1. 28
Figura 13 – Organização do MVC 30
Figura 14 – Primeiro protótipo de hardware do terminal de controle de acesso, montado em uma matriz de contatos. 31
Figura 15 – Segundo protótipo de hardware do terminal, construído em uma placa de circuito impresso. 32
Figura 16 – Protótipo final de hardware do terminal, dividido em módulos interno e externo. 33
Figura 17 – Modelagem 3D da caixa e dos componentes do módulo externo do terminal. 34
Figura 18 – Partes da caixa externa produzidas por meio do método de impressão 3D. 34
Figura 19 – Demonstração do acesso à interface do administrador por meio da comunicação Bluetooth entre um smartphone e o módulo HC-05. . 35
Figura 20 – Modelagem 3D da base e dos componentes do módulo interno do terminal. 35
Figura 21 – Partes do módulo interno. 36
Figura 22 – Sistema de controle de acesso comercial instalado no LASID. . . . 36
Figura 23 – Diagrama de entidades-relacionamentos do Projeto Ubique. 37
Figura 24 – Diagrama de entidades-relacionamentos do sistema de controle de acesso. 38
Figura 25 – Diagrama de blocos do software do terminal de controle de acesso. 41
Figura 26 – Diagrama de blocos do software do servidor web. 44
Figura 27 – Diagrama das principais classes do servidor web. 45

LISTA DE CÓDIGOS

1 Exemplo de mensagem JSON para cadastro de permissão. 39
2 Exemplo de mensagem de envio de permissão em JSON com cabeçalho HTTP. 40
3 Exemplo de mensagem enviada ao nodo *gateway* para ser retransmitida ao servidor. 42
4 Exemplo de mensagem enviada ao servidor, retransmitida pelo nodo *gateway*. 42
5 Exemplo de mensagem enviada ao nodo *gateway*, a ser retransmitida para o nodo fim. 43
6 Arquivo de configuração do software Apache para o servidor web. . . . 47
7 Arquivo de configuração do adaptador de rede USB para conexão com a rede Ubique. 47

LISTA DE ABREVIATURAS E SIGLAS

BOCC	*Building Operations Command Center*
CI	Centro de Informática
CPU	*Central Processing Unit*
CSS	*Cascading Style Sheets*
DHCP	*Dynamic Host Configuration Protocol*
HTML	*Hypertext Markup Language*
HTTP	*Hypertext Transfer Protocol*
HVAC	*Heating, Ventilation and Air Conditioning*
I2C	*Inter-Integrated Circuit*
IEEE	*Institute of Electrical and Electronic Engineers*
IoT	*Internet of Things*
IP	*Internet Protocol*
JSON	*JavaScript Object Notation*
LASID	Laboratório de Sistemas Digitais
LCD	*Liquid-Crystal Display*
LED	*Light Emitting Diode*
LUMO	Laboratório de Computação Ubíqua e Móvel
MVC	*Model-View-Controller*
PCB	*Printed Circuit Board*
PHP	*PHP Hypertext Preprocessor*
PLA	*Polylactic Acid*
RFID	*Radio-Frequency Identification*
RTC	*Real-Time Clock*
SSH	*Secure Shell*
TCP/IP	*Transmission Control Protocol/Internet Protocol*
UFPB	Universidade Federal da Paraíba
USB	*Universal Serial Bus*
WPA2	*Wi-Fi Protected Access II*
WPS	*Wi-Fi Protected Setup*

SUMÁRIO

1 INTRODUÇÃO 12
1.1 Objetivos 13
1.2 Estrutura do trabalho 14

2 FUNDAMENTAÇÃO TEÓRICA 15
2.1 Edifícios Inteligentes 15
2.2 Internet das Coisas 17

3 PROJETO UBIQUE 19
3.1 Arquitetura Ubique 20

4 MATERIAIS E MÉTODOS 22
4.1 Dispositivo de comunicação sem fios 22
4.2 Tecnologia de identificação pessoal 23
4.3 Dispositivos de hardware adicionais 24
4.4 Projeto e construção do hardware 24
4.5 *Frameworks* de software 29
4.5.1 Arduino 29
4.5.2 Laravel 30

5 HARDWARE DO CONTROLE DE ACESSO 31

6 ARQUITETURA E VALIDAÇÃO 37
6.1 Modelagem do software 37
6.2 Comunicação 39
6.3 Software do terminal de acesso 40
6.4 Software do servidor web 43
6.5 Configuração da rede e do servidor 45
6.6 Testes e validação 47

7 CONCLUSÕES E TRABALHOS FUTUROS 50

REFERÊNCIAS 52

APÊNDICE A – DOC. DE REQUISITOS DO SIST. DE ACESSO 55

APÊNDICE B – CIRCUITO DO TERMINAL DE ACESSO 58

1 INTRODUÇÃO

Os conceitos relacionados à área de edifícios inteligentes (do inglês, *smart buildings*) não são recentes. Há várias décadas, pesquisas têm sido desenvolvidas com o intuito de implementar e melhorar tecnologias utilizadas nas edificações residenciais e comerciais, tais como: sistemas de monitoramento, detecção e combate a incêndio, sistemas de ventilação e refrigeração, entre outros. Tais sistemas já são utilizados no Pentágono dos Estados Unidos, antes mesmo de terem se popularizado (GREEN, 2009). Contudo, novos estudos continuam surgindo para as mais diversas aplicações relacionadas aos *smart buildings*.

Nos últimos anos, as mudanças climáticas têm se destacado como enaltecedor da necessidade de pesquisas na área de automação e edifícios inteligentes (GREEN, 2009; BUCKMAN; MAYFIELD; BECK, 2014). Devido à maior conscientização que tem sido promovida, boa parte dos estudos possuem como foco a melhoria da eficiência energética. Essa melhoria dá-se tanto por meio do uso de equipamentos que consomem menos energia, quanto no controle eficiente dos equipamentos já existentes.

Ao mesmo tempo, a Internet das Coisas (do inglês, *Internet of Things* – IoT) está experimentando um crescimento exponencial. Tecnologias de comunicação estão se tornando cada vez mais ubíquas em eletrodomésticos e outros aparelhos eletrônicos do nosso cotidiano, provendo-os conexão com outros equipamentos, inclusive por meio da Internet. Outro fator que pode ter contribuído para o crescimento desta área é a maior disponibilidade de microcontroladores que possuem tecnologias de conectividade sem fios, atrelados à possibilidade de programá-los utilizando o *framework* de desenvolvimento *open source* Arduino, amplamente utilizado para a construção de dispositivos inteligentes. Assim, fica evidente que a Internet das Coisas pode se tornar indispensável no desenvolvimento e difusão de tecnologias para tornar os edifícios mais inteligentes.

Nesse contexto, este trabalho apresenta a concepção e o desenvolvimento do Projeto Ubique: uma rede de sensores sem fios, nos moldes da Internet das Coisas, com o intuito de prover inteligência à estrutura do Centro de Informática (CI) da Universidade Federal da Paraíba (UFPB). A partir de vivências diárias dos discentes e docentes foram identificados dois problemas principais no CI, que podem ser amenizados por meio do uso de tecnologias inteligentes. O primeiro é a falta de controle de acesso às salas de aula e laboratórios, acarretando na vulnerabilidade dos equipamentos neles dispostos. O segundo é a falta de gestão dos aparelhos de ar condicionado, que ora são encontrados funcionando sem a presença de qualquer

pessoa no ambiente (inclusive durante os fins de semana), ora não são ligados durante as aulas devido à ausência dos controles remotos.

Atualmente, o Centro dispõe de pontos de rede cabeada em vários ambientes com acesso à Internet aberta a todos os usuários. Como a disposição e a quantidade desses pontos de rede não serão compatíveis com as localizações mais apropriadas para os dispositivos do Projeto Ubique, e visto que a expansão dessa rede cabeada seria custosa e burocrática, a adoção de tecnologias de comunicação sem fios é uma possibilidade. No CI há diversas redes Wi-Fi disponíveis, entretanto, por serem de uso comum e apresentarem configurações diversas de acesso e segurança, tais redes não poderiam ser adotadas pelo Ubique, que demanda, assim, uma rede sem fios exclusiva para seus dispositivos.

Como forma de validação da arquitetura proposta, foi desenvolvido um sistema de controle de acesso eletrônico para os laboratórios e salas de aula do Centro. Tal sistema é composto por um servidor, que armazena as informações dos ambientes, usuários habilitados e nodos da rede, e pelos terminais de controle de acesso, que são um tipo de nodo da rede de sensores sem fios do Ubique. Nesse sistema, cada usuário é identificado por meio de um cartão inteligente que utiliza tecnologia de rádio frequência (do inglês, *Radio-Frequency Identification* – RFID), cadastrado no servidor e armazenado nos terminais nos quais o usuário possui permissão de acesso.

Além do controle de acesso aos seus ambientes, o Projeto Ubique prevê o uso de tecnologias voltadas à eficiência energética do Centro de Informática, tais como o monitoramento e controle dos aparelhos de ar condicionado, de iluminação, computadores e outros dispositivos que possam demandar um gasto significativo de energia. Esses projetos e suas pesquisas estão sendo desenvolvidos no Laboratório de Computação Ubíqua e Móvel – LUMO.

1.1 OBJETIVOS

O principal objetivo deste trabalho é apresentar a arquitetura e o processo de desenvolvimento do Projeto Ubique, usando como estudo de caso o Centro de Informática. Como forma de validação da rede de sensores sem fios Ubique, foi desenvolvido um sistema de controle de acesso. Pode-se identificar como objetivos específicos:

- Desenvolvimento de protótipo de terminal de controle de acesso;
- Desenvolvimento do servidor web para gerenciamento da rede de sensores;
- Identificação das questões de segurança inerentes ao projeto.

1.2 ESTRUTURA DO TRABALHO

Este trabalho está dividido em 7 capítulos. No Capítulo 2 são apresentados os conceitos de edifícios inteligente e Internet das Coisas. No Capítulo 3 é abordado Projeto Ubique e sua arquitetura. No Capítulo 4 são expostos os materiais e métodos utilizados para desenvolver o sistema de controle de acesso no âmbito do Projeto Ubique. O desenvolvimento do hardware do terminal de controle de acesso é apresentado no Capítulo 5, enquanto no Capítulo 6 são apresentados a arquitetura do software do Ubique, o desenvolvimento do software do terminal, o desenvolvimento, instalação e configuração do servidor web e da rede Wi-Fi, e os testes e validações executados. Por fim, o Capítulo 7 apresenta as conclusões, as questões de segurança identificadas e os possíveis trabalhos futuros que podem ser desenvolvidos, tanto no contexto do Projeto Ubique quanto no próprio sistema de controle de acesso.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo são abordados dois conceitos fundamentais, que nortearam o desenvolvimento deste trabalho: os edifícios inteligentes, frutos da modernização das edificações por meio do uso de tecnologias inteligentes; e a Internet das Coisas, composta por objetos do cotidiano dotados de tecnologias de comunicação.

2.1 EDIFÍCIOS INTELIGENTES

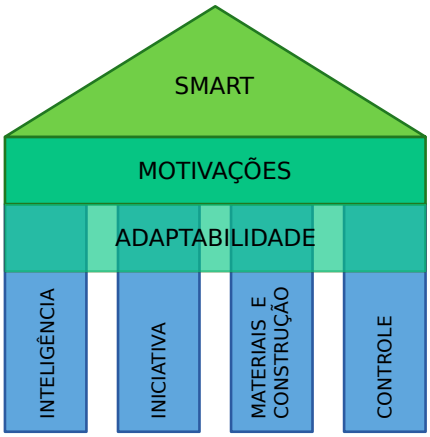
O projeto e execução de construções civis, sejam elas residenciais ou comerciais, tem sofrido alterações ao longo do tempo de forma a melhorar sua eficiência. As motivações para tais mudanças estão quase sempre relacionadas à adição de valor às construções, valor esse em termos de custos menores ao longo do tempo, conforto e satisfação daqueles que usufruem da estrutura, e que depende do propósito da edificação (SHABHA; CABA, 2006, 2008 apud BUCKMAN; MAYFIELD; BECK, 2014).

Tendo em vista a magnitude dos custos de operação (tais como manutenções e consumo energético) de um edifício em relação ao seu custo inicial de construção, Buckman, Mayfield e Beck (2014) apontam três principais motivações para sua evolução: longevidade, isto é, a capacidade de manter seu valor ao longo do tempo quando submetido a mudanças externas e de condições de uso; conforto e satisfação de seus ocupantes; e a eficiência energética, que não agrega valor diretamente, mas que busca atender a requisitos legais e minimizar o impacto no meio ambiente.

Nesse contexto, e tomando como base as tentativas de diversos autores em definir o conceito de inteligência no contexto das edificações, Buckman, Mayfield e Beck (2014) apresentam sua definição de *smart buildings* como sendo aqueles que integram inteligência, autonomia, controle, materiais e construção em um sistema único de edificação. A adaptabilidade é tida como fundamental, a fim de atender às motivações para a evolução das construções: eficiência energética, longevidade, conforto e satisfação. Os autores ilustram a relação entre os principais conceitos na Figura 1.

Sinopoli (2010) afirma que o conceito de edifícios inteligentes está relacionado à instalação e ao uso de diversos sistemas prediais, tais como automação, segurança, telecomunicações, entre outros. O autor ressalta também que, diferente do método tradicional, no qual o projeto, instalação e operação de cada sistema é independente, nos edifícios inteligentes o projeto e o funcionamento dos sistemas são integrados

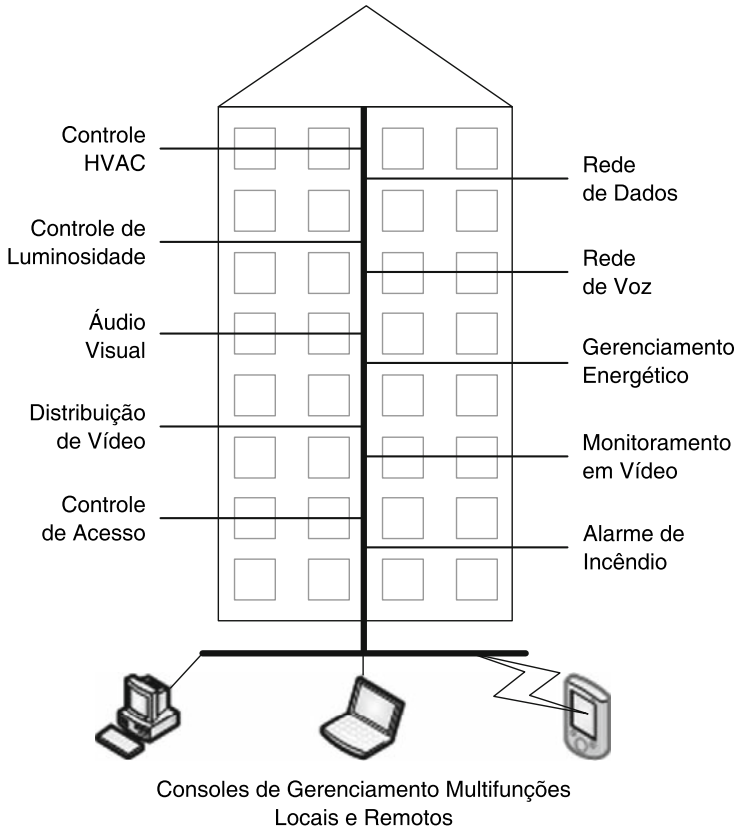
Figura 1 – Características de um edifício inteligente.



Fonte: Buckman, Mayfield e Beck (2014), adaptada pelo autor.

por meio do compartilhamento de estruturas de comunicação, bancos de dados e protocolos (integração horizontal). Isso permite também que subsistemas possam ser utilizados por sistemas maiores de gerenciamento (integração vertical). Na Figura 2, é exibido um exemplo de integração de diferentes sistemas prediais.

Figura 2 – Integração entre diferentes sistemas prediais.



Fonte: Sinopoli (2010), adaptada pelo autor.

A adaptabilidade, de acordo com Buckman, Mayfield e Beck (2014), é um

conceito chave na definição de edifícios inteligentes. Este requisito de qualidade de um sistema inteligente está diretamente relacionado ao uso de informações internas e externas obtidas de diferentes fontes para preparar a construção para um evento particular, antes que ele aconteça. Alguns desses eventos são: mudanças climáticas, mudanças nas taxas de ocupação e diferentes percepções de conforto das pessoas, de acordo com a hora do dia e a época do ano; tudo isso considerando a eficiência energética e a satisfação dos seus usuários.

Um dos edifícios governamentais mais famosos do mundo, o Pentágono dos Estados Unidos, adotou tecnologias inteligentes antes mesmo da sua popularização. No ano de 1997 foi concebida uma central de comando para automação predial do Pentágono, o *Building Operations Command Center* (BOCC) (SNOONIAN, 2003). O BOCC provê aos supervisores acesso ao estado de funcionamento de todos os sistemas, além de informações relativas à segurança, qualidade do ambiente interno, eficiência operacional e dados do sistema fotovoltaico, concentrando dados sobre a quantidade de energia gerada por cada célula. Os sistemas do BOCC agregam 16 funções de automação e monitoramento, tais como: detecção, combate e segurança contra incêndios, aquecimento, ventilação e refrigeração – HVAC (do inglês, *heating, ventilation and air conditioning*), luminosidade, energia e controle de acesso. Essas funções visam manter os sistemas operacionais com o mínimo de interação com os usuário (GREEN, 2009). O maior desafio foi a integração desses diversos sistemas em um só, de forma a melhorar o gerenciamento, oferecer suporte a missões críticas, não apresentar um único ponto de falha, durar décadas – e ter um bom custo-benefício (JOHNSON CONTROLS, 2003 apud GREEN, 2009).

Um dos momentos críticos do sistema aconteceu durante os ataques do dia 11 de setembro de 2001, quando o avião do voo 77 da American Airlines foi sequestrado e direcionado ao Pentágono, causando a morte de 189 pessoas e US$ 501 milhões em prejuízos. Na ocasião, os sistemas de automação foram utilizados para controlar passagens de ar para conter a fumaça e suprimir o incêndio (cortando o fornecimento de oxigênio), além de diagnosticar defeitos e direcionar reparos ao sistema hidráulico de combate ao incêndio. Graças aos sistemas de automação e aos reparos, os danos foram contidos, diversos setores críticos foram poupados, e as operações no Pentágono foram retomadas no dia seguinte. (SNOONIAN, 2003).

2.2 INTERNET DAS COISAS

A Internet tem crescido nos últimos 50 anos, sendo inicialmente uma rede com poucos componentes (nodos) voltada para pesquisas, e tornando-se uma rede mundial e pervasiva, atendendo mais de um bilhão de usuários e possuindo vários

dispositivos interconectados de forma complexa. A miniaturização e a redução do custo de produção de dispositivos eletrônicos contribuiu para o surgimento de uma nova área: a dos objetos inteligentes, compostos de coisas do nosso dia-a-dia munidas de pequenos dispositivos eletrônicos que proveem inteligência e conexão com outros objetos e/ou com a Internet (KOPETZ, 2011). Desse forma, a Internet das Coisas (IoT) pode ser vista como a ubiquidade desses objetos inteligentes no nosso cotidiano, isto é, a (quase) inexistência de uma diferença entre as nossas ações e a operação desses objetos, integrados de forma quase imperceptível (MENDEZ; PAPAPANAGIOTOU; YANG, 2018).

A novidade trazida pela IoT não é o surgimento desses dispositivos eletrônicos inteligentes e interconectados, mas sim a perspectiva de que haja bilhões ou até mesmo trilhões de objetos inteligentes, que trazem consigo novos problemas técnicos e sociais devido à sua abrangência, tais como autenticidade, segurança (física e da informação) e privacidade (KOPETZ, 2011).

Diversas áreas podem ser beneficiadas com a aplicação da Internet das Coisas. Algumas delas são a logística, eficiência energética, indústria, área médica, e também a segurança física, que é um dos focos deste trabalho. Nesta última área, destaca-se o uso de sistemas de controle de acesso eletrônicos, seja em locais públicos como universidades, bibliotecas, aeroportos, como também em locais privados como hotéis, hospitais, condomínios e até mesmo residências particulares.

Dentre os problemas técnicos característicos da IoT, pode-se identificar: a integração com a Internet, que em algumas ocasiões não se dá por meio de protocolos convencionais devido às limitações energética e de processamento; nomeação e identificação, de forma a rotular cada objeto inteligente de forma única e universal, além dele poder assumir papéis diferentes ou até mesmo mais de um papel simultaneamente; e, ainda mais importante, a segurança dos dispositivos e da informação.

A área de segurança na IoT possui diversas peculiaridades, como mostram Hossain, Hasan e Skjellum (2017). A principal delas é o baixo poder de processamento disponível nos mais variados dispositivos (nodos) utilizados, caracterizado por CPUs mais lentas, menos memória (tanto de processamento quanto de armazenamento), e em alguns casos, limitações energéticas (nos casos de nodos alimentados por baterias). Dentre as soluções de segurança atuais, os autores ressaltam a dificuldade em encontrar uma que acomode essa diversidade de dispositivos heterogêneos.

Outras características inerentes à área incluem a baixa taxa de transmissão de dados, dificuldade em aplicar atualizações de *firmware* nos dispositivos, mobilidade e volatilidade da conexão dos nodos e a diversidade de tecnologias e sensores utilizados. Tais características dificultam o desenvolvimento de soluções de software compatíveis com todos eles (MENDEZ; PAPAPANAGIOTOU; YANG, 2018).

3 PROJETO UBIQUE

O Centro de Informática dispõe de 64 ambientes fechados, dos quais 10 são salas de aula, 8 são laboratórios de graduação (Informática e Eletricidade) e 11 são laboratórios de pesquisa. Algumas salas de aula dispõem de aparelhos de ar condicionado e projetores digitais. Os laboratórios de Informática, além desses itens, apresentam também *desktops* e infraestrutura de cabeamento lógico. Já os laboratórios de Eletricidade são equipados com dispositivos de instrumentação de custo elevado, tais como: osciloscópios, geradores de sinais, fontes de tensão ajustáveis etc.

Atualmente, o controle de acesso a esses ambientes não é padronizado. Nas salas de aula e nos laboratórios de Informática, tal controle é inexistente, e as portas ficam destrancadas na maior parte do tempo. Já o controle dos laboratórios de Eletricidade é feito pelos poucos professores que utilizam estes ambientes.

Tal situação torna o processo de utilização desses ambientes burocrático e dependente do contato com os responsáveis por suas chaves. Não é raro presenciar situações em que professores estejam sujeitos à disponibilidade dos técnicos administrativos para abrir os laboratórios, ou que a chave tenha sido cedida para um terceiro. Alunos que queiram desenvolver projetos nos laboratórios de Eletricidade também são limitados à presença dos professores no Centro para emprestar e devolver a chave. Os mesmos problemas são enfrentados pelos funcionários do setor de limpeza e demais que necessitem utilizar os ambientes compartilhados.

Outro problema enfrentado não só pelo Centro de Informática, mas também pela gestão da UFPB, é o alto consumo de energia elétrica de todos os setores, sendo os aparelhos de ar condicionado alguns dos maiores contribuintes. Sua operação é muitas vezes feita de forma indevida, seja por meio de configuração inadequada ou até mesmo devido ao fato de os aparelhos serem mantidos funcionando quando não há ninguém presente nos ambientes.

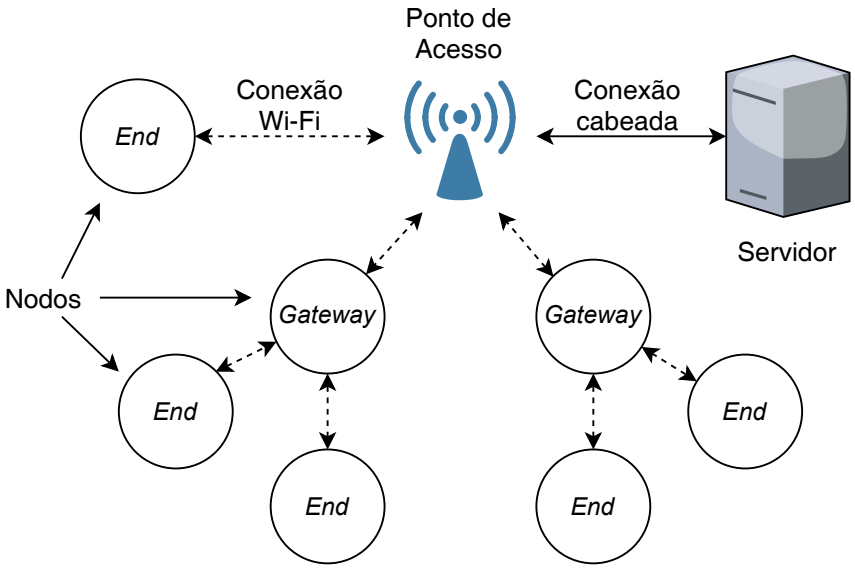
Nesse contexto, é evidente a necessidade do desenvolvimento de soluções para os problemas apresentados. Inicialmente, como proposta de solução, foi concebido um sistema de controle de acesso eletrônico para os ambientes de uso comum e laboratórios de pesquisa do Centro, o CI-Access, no âmbito da disciplina de Computação Pervasiva. Entretanto, após constatada a demanda de mecanismos para melhorar a eficiência energética do CI, percebeu-se que poderiam ser desenvolvidos dispositivos que não só atendesse essa demanda, mas que também compartilhassem a mesma estrutura de comunicação do sistema de controle de acesso. Surgiu assim o Projeto Ubique, uma rede de sensores sem fios que oferece suporte a diferentes

sistemas de inteligência para edifícios como o Centro de Informática.

3.1 ARQUITETURA UBIQUE

A rede de sensores sem fios do Ubique é composta por nodos, pontos de acesso e um servidor, como representado pela Figura 3. Os nodos que compõem a estrutura principal da rede (*backbone*) são dispositivos embarcados dotados de tecnologia de comunicação sem fios compatível com o protocolo Wi-Fi (IEEE 802.11). Sistemas embarcados que utilizem outras tecnologias de comunicação podem ser integrados à rede, por meio de um nodo concentrador que possua comunicação Wi-Fi. Além das funções de comunicação, os nodos são responsáveis pelo sensoriamento e atuação nos ambientes em que estão instalados. No sistema de controle de acesso, eles são terminais instalados nas entradas dos ambientes, cuja função é ler os cartões inteligentes e controlar o mecanismo de trava da porta, que pode ser um eletroímã ou um fecho elétrico.

Figura 3 – Arquitetura da rede de sensores sem fios do Projeto Ubique.



Fonte: Próprio autor.

O servidor, por sua vez, é a central de controle e armazenamento de informação. Por meio dele é possível gerenciar os dados relativos à operação de cada sistema, bem como ter acesso aos eventos e outras informações reportadas pelos nodos da rede. Pode-se também atualizar as configurações de cada nodo e emitir comandos para atuação direta nos ambientes em que estão instalados, ligando ou desligando um aparelho de ar condicionado, por exemplo.

Os pontos de acesso são dispositivos que permitem a conexão entre os nodos e o servidor por meio de uma rede sem fios. Entretanto, o Ubique oferece também

a possibilidade de os nodos serem configurados em modo híbrido, no qual podem também disponibilizar uma rede sem fios própria, intermediando assim a conexão de outros nodos com o servidor. Tais nodos híbridos são chamados *gateway*, enquanto os demais são chamados nodos fim (do inglês, *end nodes*). Os nodos *gateway* foram concebidos com o intuito de aumentar o alcance da rede de sensores sem o uso de pontos de acesso, visto que cada um destes necessita de uma conexão de rede cabeada. Ressalta-se que, para um nodo operar no modo *gateway*, ele deve estar conectado ao ponto de acesso, limitando assim o encadeamento de nodos a 2 níveis (um nível *gateway* e outro nível fim).

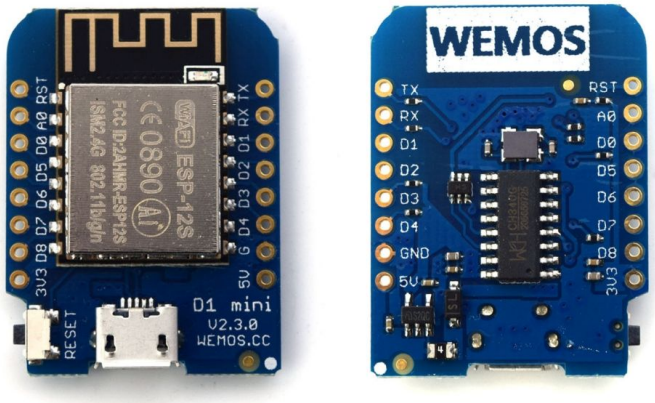
4 MATERIAIS E MÉTODOS

Neste trabalho é apresentado o desenvolvimento de duas partes principais: um terminal de controle de acesso, responsável por realizar a leitura dos cartões inteligentes, controlar a abertura da porta do ambiente e registrar eventos; e um servidor web, para gerenciar as entidades e os nodos da rede, registrar os eventos e emitir relatórios. Dessa forma, este capítulo apresenta os componentes e técnicas utilizados na construção do terminal de acesso, bem como as ferramentas de software adotadas para o desenvolvimento do sistema embarcado e do servidor web.

4.1 DISPOSITIVO DE COMUNICAÇÃO SEM FIOS

Um dos dispositivos embarcados com popularidade cada vez maior na área da Internet das Coisas é o microcontrolador Espressif ESP8266EX, que possui conectividade Wi-Fi integrada e pode ser facilmente programado utilizando o *framework* de desenvolvimento Arduino e suas inúmeras bibliotecas de software disponíveis. Esses fatores, somados ao custo dos módulos que utilizam esse microcontrolador e sua maior disponibilidade no mercado nacional, fundamentam a escolha dele e, consequentemente, da tecnologia Wi-Fi como base para os dispositivos do Ubique, em detrimento de outras alternativas de sensoriamento sem fios como ZigBee, LoRa e Bluetooth. O módulo de desenvolvimento utilizado neste trabalho foi o WEMOS D1 Mini v2, apresentado na Figura 4, devido ao seu tamanho reduzido e por oferecer conectividade USB para programação.

Figura 4 – Módulo de desenvolvimento WEMOS D1 Mini.



Fonte: WEMOS Wiki[1], adaptada pelo autor.

[1] Disponível em: <https://wiki.wemos.cc/products:retired:d1_mini_v2.3.0>. Acesso em: 1 jun 2018.

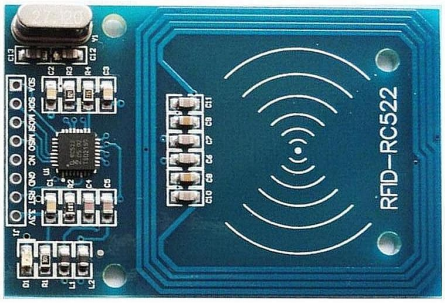
4.2 TECNOLOGIA DE IDENTIFICAÇÃO PESSOAL

Para a identificação pessoal, necessária para o sistema de controle de acesso, há diversas soluções disponíveis, dos mais variados níveis de complexidade e tecnologias utilizadas. Uma das mais simples, a senha numérica, depende da memória do usuário, que tenderá a usar uma numeração conhecida e de fácil acesso (datas de aniversário, números de telefone etc), ou usar a mesma senha em outros sistemas, além de poder ser observada e reproduzida por terceiros facilmente.

Outra forma de autenticação mais sofisticada e que está se popularizando em *smartphones* é por meio do uso da impressão digital. Contudo, essa tecnologia apresenta dois problemas principais: um deles é que a captura e o processamento da digital dependem da qualidade dos sensores utilizados e das condições dos dedos do usuário (sujeira, umidade, ferimentos, etc); o segundo diz respeito às críticas quanto a sua segurança, como mostrado por O'Gorman (2003), Kirkland (2013), e no site XDA Developers (2015). Tais autores ressaltam a impossibilidade de alterar essa forma de autenticação em caso de comprometimento.

Nesse sentido, outro método popular de identificação pessoal é por meio de cartões inteligentes. Cada usuário do sistema possui um cartão eletrônico que contém dados que o identificam unicamente e não são facilmente sujeitos à cópia (tanto por questões físicas quanto por mecanismos de segurança intrínsecos ao cartão). Esse, portanto, foi o método de autenticação adotado pelo sistema de controle de acesso desenvolvido neste trabalho, particularmente na sua versão RFID de 13,56 MHz, por motivos similares à adoção do ESP8266 (facilidade de integração, custo e disponibilidade no mercado). O módulo utilizado foi o RFID-RC522, mostrado na Figura 5, baseado no microcontrolador NXP MFRC522.

Figura 5 – Módulo de desenvolvimento RFID-RC522.



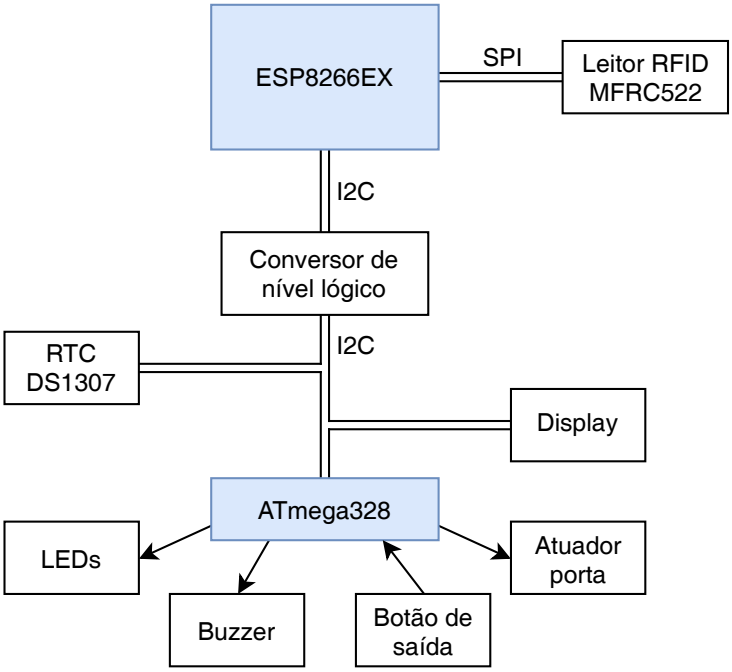
Fonte: Lelong.my[2], adaptada pelo autor.

[2] Disponível em: <https://www.lelong.com.my/rfid-rc522-module-tonsin-172083032-2017-12-Sale-P.htm>. Acesso em: 1 jun 2018.

4.3 DISPOSITIVOS DE HARDWARE ADICIONAIS

Além desses dois componentes principais, foram utilizados também no hardware dos terminais de controle de acesso um módulo de relógio em tempo real (do inglês, *Real-Time Clock* – RTC) e um microcontrolador Atmel ATmega328p (embarcado em uma placa Arduino Pro Mini) para prover mais pinos de entrada e saída para o ESP8266. Como interface para os usuários foram adotados LEDs verde e vermelho, um *buzzer*, e um *display* LCD alfanumérico de 16×2 caracteres, que exibe informações de data e hora e que, embora não implementado na versão atual do software, pode ser utilizado também para mostrar informações relevantes do ambiente. Um diagrama conceitual da arquitetura do hardware do terminal pode ser visto na Figura 6. Como o ESP8266 utiliza uma tensão de 3,3 V para representar o nível lógico alto, enquanto os demais periféricos (exceto o leitor RFID) utilizam 5 V, utilizou-se um conversor de nível lógico.

Figura 6 – Arquitetura de hardware do terminal de controle de acesso.



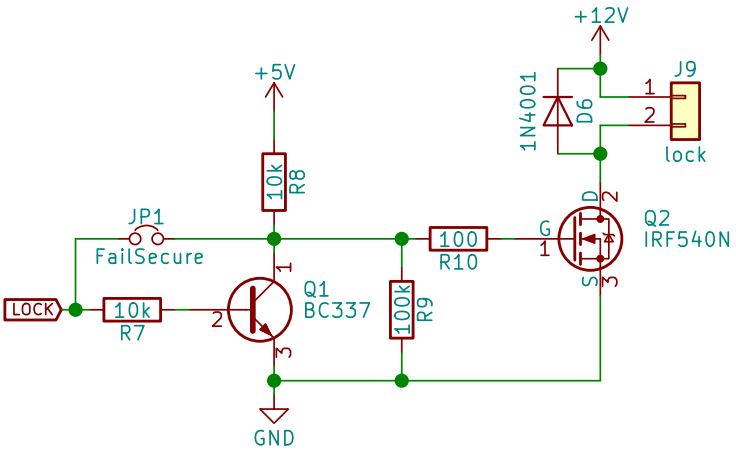
Fonte: Próprio autor.

4.4 PROJETO E CONSTRUÇÃO DO HARDWARE

Para validar o sistema de controle de acesso, foram construídos diversos protótipos do terminal, utilizando várias ferramentas, processos e técnicas que serão apresentadas a seguir.

O primeiro passo na construção de hardware consiste no projeto do circuito ou esquema elétrico. É nele que são especificados os componentes, as conexões elétricas entre seus terminais e as entradas e saídas do circuito, de forma a cumprir uma função desejada. Uma parte do circuito elétrico do terminal de acesso é exibida na Figura 7, na qual o sinal para destravar a porta (LOCK, à esquerda) é utilizado para comandar o atuador da trava (J9, à direita).

Figura 7 – Parte do circuito elétrico do terminal de controle de acesso.



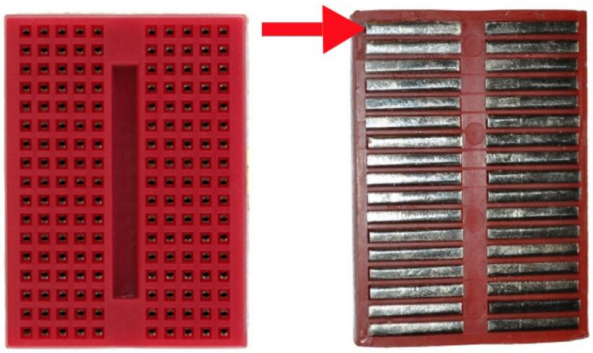
Fonte: Próprio autor.

Há diversas ferramentas de software que auxiliam nesse processo, não só no desenho do esquema elétrico, como também no projeto da placa de circuito impresso, que será apresentada mais adiante. Neste trabalho foi utilizado o pacote KiCad EDA[3], escolhido principalmente por ser gratuito, *open source* e multiplataforma. Além dos softwares de desenho elétrico e da PCB, o KiCad fornece também uma extensa biblioteca de símbolos elétricos e de componentes físicos, descritos pelas suas características (*footprints*). Fornece também ferramentas para que possam ser especificados novos símbolos e componentes.

O primeiro protótipo do hardware foi construído em uma matriz de contatos, conhecida também como *protoboard*. É composto por uma base plástica, com pequenas perfurações organizadas em linhas e colunas e conectadas internamente por terminais metálicos, formando assim pontos de conexão elétrica. A montagem de circuitos é feita ao encaixar e conectar componentes e fios em diferentes pontos de conexão. As *protoboards* são utilizadas para a construção de protótipos iniciais e circuitos temporários, haja vista a facilidade de inserir novos componentes e realizar modificações no circuito elétrico. Na Figura 8 é possível ver os dois lados de uma *protoboard*, com destaque para os terminais metálicos no verso, que conectam os furos de uma mesma linha.

[3] <http://kicad-pcb.org/>

Figura 8 – Frente e verso de uma matriz de contatos.



Fonte: Sparkfun[4].

Uma vez validado o circuito montado na matriz de contatos, pode-se construir uma versão permanente em uma placa de circuito impresso. Também conhecida como PCB (do inglês, *printed circuit board*), na sua versão mais simples é composta por um substrato de material isolante (geralmente fibra de vidro ou fenolite) com uma ou duas de suas faces cobertas por uma folha de cobre. Para formar os circuitos, retira-se o cobre de certas áreas, com o intuito de formar trilhas e áreas de conexão entre os diferentes terminais dos componentes. Estes, por sua vez, são conectados ao cobre por meio de um processo de solda, que utiliza uma liga de estanho e chumbo.

As placas de circuito impresso podem ser fabricadas por meio de diversos processos, dentre eles o método de fresagem. Neste projeto, as PCBs foram fabricadas utilizando uma fresadora LPKF ProtoMat S42, disponibilizada no Laboratório de Sistemas Digitais (LASID). Um modelo dessa fresa pode ser visto na Figura 9. Nela, a placa virgem é afixada em uma mesa coordenada, enquanto as ferramentas rotativas de desbaste do cobre, perfuração e corte do substrato são movimentadas pela cabeça de fresagem. Para operá-la, são utilizados softwares específicos, que podem escolher as ferramentas mais apropriadas a serem utilizadas e calculam as rotas que elas têm que percorrer a partir de arquivos de fabricação gerados pelo programa de projeto de PCB. Após a fresagem, foi aplicado um verniz de uso geral para evitar a oxidação do cobre.

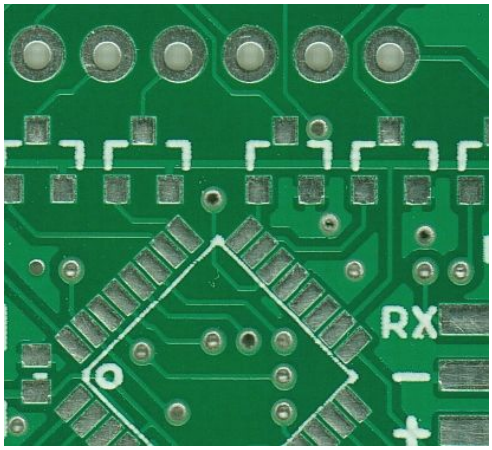
Na Figura 10 pode-se ver parte de uma placa de circuito impresso, produzida industrialmente e revestida de verniz isolante verde. As partes prateadas e em verde claro são as áreas de cobre, dispostas sobre o substrato em verde escuro. Os terminais dos componentes podem ser soldados apenas nas áreas prateadas (estanhadas). Além do verniz, o processo industrial permite a aplicação de uma camada adicional de tinta, contendo os desenhos dos componentes utilizados, em branco na mesma figura.

[4] Disponível em: <https://learn.sparkfun.com/tutorials/how-to-use-a-breadboard>. Acesso em: 4 jun 2018.

Figura 9 – Fresadora LPKF ProtoMat S42.

Fonte: LPKF (2006), adaptada pelo autor.

Figura 10 – Parte de uma placa de circuito impresso.

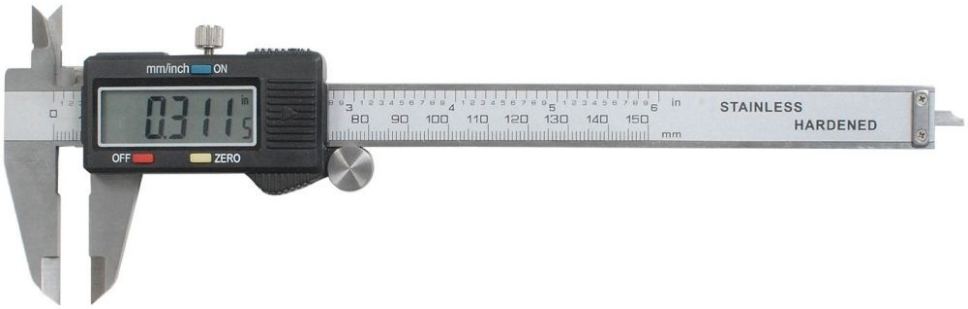


Fonte: Sparkfun[5].

Para proteger as placas de circuito impresso, seus componentes, conexões e módulos de hardware utilizados, foi confeccionada uma caixa plástica personalizada. Seu projeto foi realizado utilizando o software Trimble SketchUp Make 2017[6], escolhido por ser gratuito e pela experiência prévia do autor. Para projetar a caixa, foi necessário também fazer a modelagem em 3D das placas utilizadas, atentando para dimensões e localização de furos de fixação, LEDs e do *display*. Tais dimensões foram aferidas utilizando um paquímetro digital, um instrumento de medição de precisão, que pode ser visto na Figura 11.

[5] Disponível em: <https://learn.sparkfun.com/tutorials/pcb-basics>. Acesso em: 4 jun 2018.
[6] <https://www.sketchup.com/>

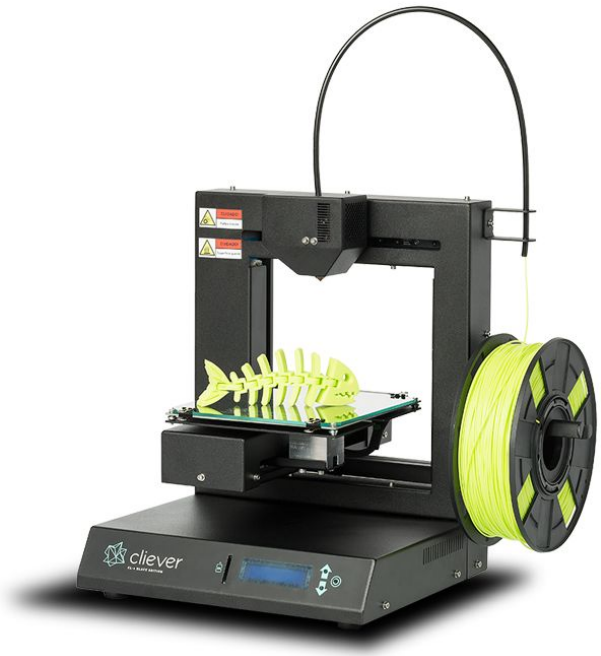
Figura 11 – Paquímetro digital.



Fonte: Parts Express[7].

A partir do projeto em 3D foi feita a confecção da caixa por meio da técnica de impressão 3D, utilizando o processo de extrusão. Nele, um filamento de material plástico é aquecido, extrudado e depositado em camadas em uma mesa coordenada. Após extrudado, o material rapidamente se solidifica e a composição das diferentes camadas forma o objeto desejado. Nesse projeto foi utilizada a impressora 3D Cliever CL1 Black Edition, apresentada na Figura 12, com filamento de material PLA (do inglês, *polylactic acid*).

Figura 12 – Impressora 3D Cliever CL1.



Fonte: Cliever[8].

[7] Disponível em: <https://www.parts-express.com/6-precision-digital-caliper--390-592>. Acesso em: 4 jun 2018.

[8] Disponível em: <https://store.cliever.com/impressoras-3d/impressora-3d-cl1-black-edition/>. Acesso em: 4 jun 2018.

4.5 *FRAMEWORKS* DE SOFTWARE

Na área da computação, um *framework* de software pode ser definido como uma estrutura genérica que pode ser adaptada para criar uma aplicação específica. São geralmente implementações de padrões de projeto, cujas classes concretas e abstratas permitem o desenvolvimento de funções mais avançadas (SOMMERVILLE, 2010). O uso de *frameworks* permite que o desenvolvimento seja mais focado na aplicação em si, e não em questões de mais baixo nível, como acesso a bancos de dados (no caso de um servidor web) ou controle das interfaces de hardware (sistema embarcado).

Neste trabalho foram utilizados dois *frameworks* de software: o Arduino, para o sistema embarcado no terminal, e o Laravel, para o servidor web.

4.5.1 ARDUINO

O Arduino é uma plataforma *open source* para dispositivos eletrônicos, que facilita o desenvolvimento de software para diversos sistemas embarcados ao abstrair detalhes de funcionamento do hardware (ARDUINO, 2018). Por possuir hardware e software *open source*, ser multiplataforma, com linguagem fácil de aprender e com placas de desenvolvimento de baixo custo, o Arduino tornou-se bastante popular, tanto para iniciantes na eletrônica e programação quanto para sistemas mais avançados, como o deste trabalho. A linguagem de programação do Arduino é baseada em C++.

Além das facilidades providas pelo *framework*, a comunidade do Arduino disponibiliza também bibliotecas dos mais variados propósitos, tanto para facilitar o uso de módulos de hardware (*displays*, sensores, motores etc), quanto para acrescentar funções de software tais como protocolos de comunicação, interface de usuário, processamento de dados e outras. Atualmente, o repositório oficial conta com mais de 1500 bibliotecas[9], todas de código aberto.

Infelizmente, como o Arduino é mais voltado para iniciantes, seu ambiente de desenvolvimento padrão é bastante simplificado, e não conta com ferramentas mais avançadas tais como versionamento, sugestão dinâmica de funções e análise de código, recursos essenciais para o desenvolvimento e gerenciamento de softwares mais complexos. Nesse cenário, foi utilizado neste projeto o PlatformIO, uma ferramenta de desenvolvimento para sistemas embarcados que gerencia automaticamente ferramentas de compilação e bibliotecas, de acordo com o hardware para o qual o software está sendo desenvolvido. Atualmente, o PlatformIO oferece suporte a 16 *frameworks* de desenvolvimento (incluindo o Arduino), 484 sistemas embarcados e mais de 5000 bibliotecas (PLATFORMIO, 2018). O PlatformIO foi utilizado na sua versão para o editor de texto Atom.

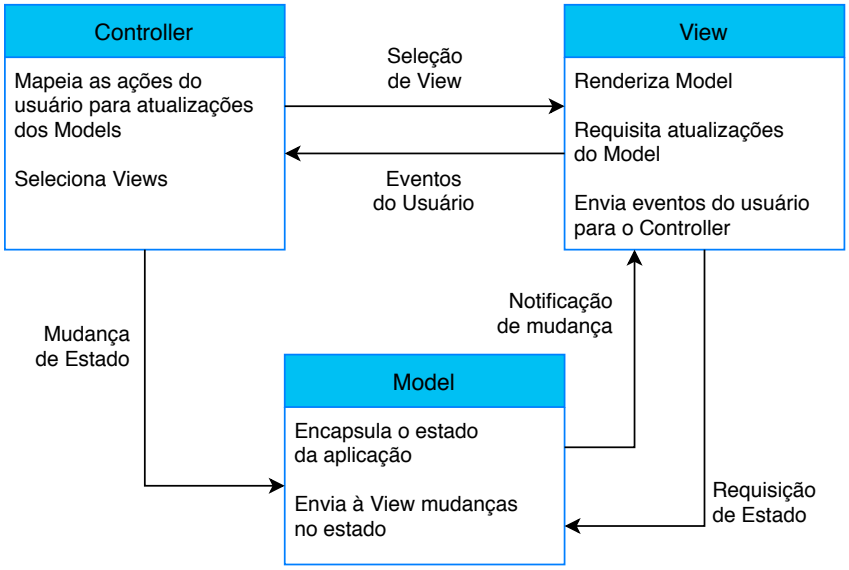
[9] Disponível em: <https://www.arduinolibraries.info/>. Acesso em: 4 jun 2018.

4.5.2 LARAVEL

O Laravel é um *framework open source* de desenvolvimento para a web, baseado na linguagem PHP, criado em 2011 e atualmente na sua versão 5.6. Por padrão, ele oferece funções de roteamento[10], gerenciamento de sessões e cache, processamento de tarefas, autenticação de usuários, localização, interface para banco de dados e diversas outras (LARAVEL, 2018).

O Laravel é baseado no padrão de projeto arquitetural *Model-View-Controller* (MVC), que promove uma separação entre os dados do sistema, sua apresentação e interações, por meio de três componentes que interagem entre si: o *model*, que gerencia os dados e as operações associadas; a *view*, que define como os dados são apresentados aos usuários do sistema; e o *controller*, que gerencia as interações dos usuários e manipula as *views* e os *models* (SOMMERVILLE, 2010). A Figura 13 apresenta um resumo desse padrão de projeto.

Figura 13 – Organização das classes do padrão de projeto MVC.



Fonte: Sommerville (2010), adaptada pelo autor.

O PHP é uma linguagem voltada para o desenvolvimento da lógica da aplicação. Para a interface do usuário, foram utilizadas as linguagens HTML, CSS e JavaScript, com o auxílio do *framework* Bootstrap e da biblioteca jQuery.

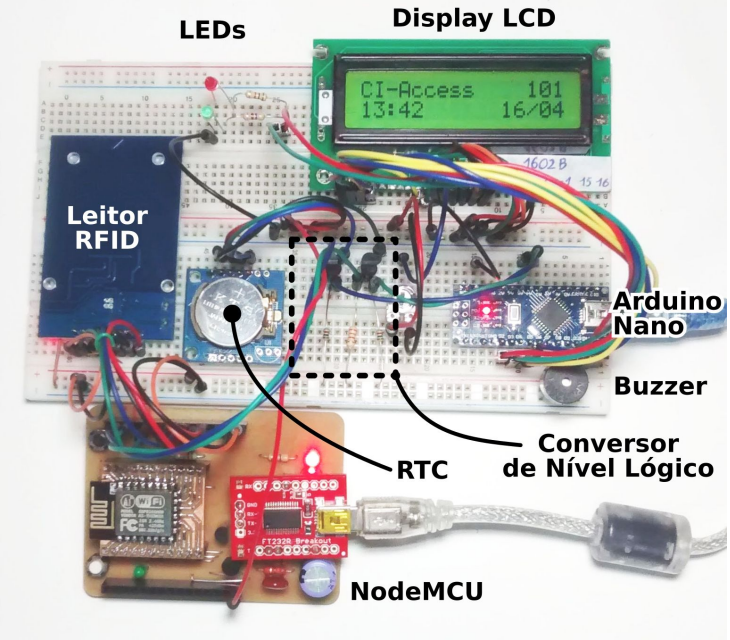
[10] No contexto de aplicações web, roteamento é o nome dado ao mecanismo utilizado para direcionar as requisições HTTP ao código responsável por tratá-las (BROWN, 2014).

5 HARDWARE DO CONTROLE DE ACESSO

O desenvolvimento do hardware do terminal de controle de acesso teve início antes da concepção do Projeto Ubique, no âmbito do projeto CI-Access da disciplina de Computação Pervasiva. Isso permitiu que os conceitos de redes de sensores sem fios pudessem ser validados, servindo assim como fundamento para a integração de outros tipos de nodos.

O primeiro protótipo de hardware foi montado em uma matriz de contatos, como dito anteriormente, permitindo o projeto e validação graduais do circuito elétrico e seus diferentes componentes. Esse protótipo inicial pode ser visto na Figura 14.

Figura 14 – Primeiro protótipo de hardware do terminal de controle de acesso, montado em uma matriz de contatos.

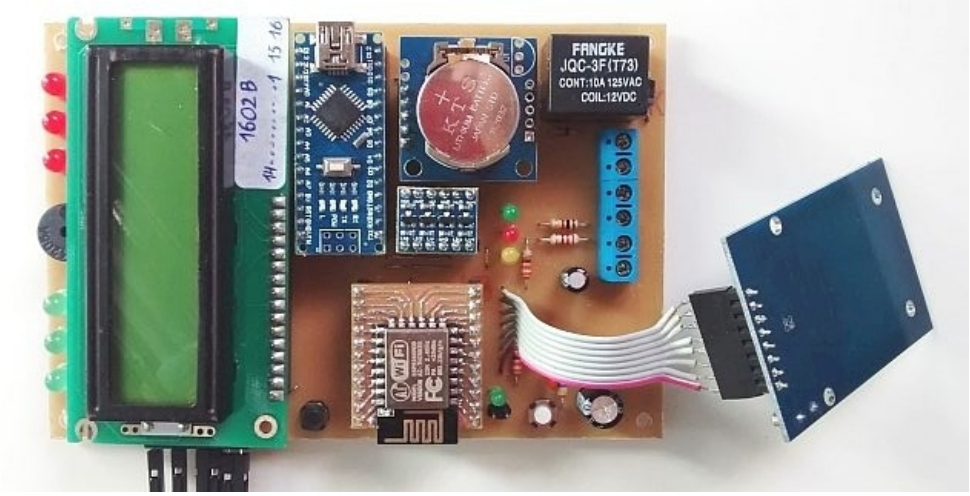


Fonte: Próprio autor.

Nessa versão inicial foram utilizadas placas de desenvolvimento diferentes da versão final: NodeMCU, com o microcontrolador ESP8266; e Arduino Nano com o ATmega328p. Utilizou-se também um *display* LCD com comunicação paralela, controlado pelo Arduino Nano.

Uma vez validado o hardware na matriz de contatos, foi estabelecido o circuito elétrico e confeccionada uma PCB, mostrada na Figura 15, que integra os componentes da placa de desenvolvimento NodeMCU, e também o atuador da trava da porta.

Figura 15 – Segundo protótipo de hardware do terminal, construído em uma placa de circuito impresso.



Fonte: Próprio autor.

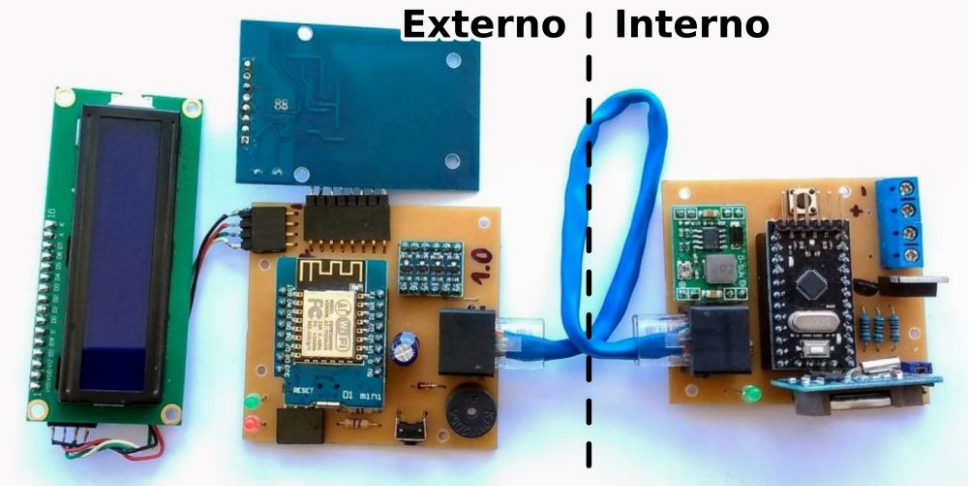
Com o passar do tempo foram concebidas algumas melhorias para o projeto do hardware. Substitui-se o circuito da NodeMCU pela placa de desenvolvimento WEMOS D1 Mini, diminuindo assim a quantidade de componentes necessários. O *display* LCD foi substituído por uma versão com interface de comunicação I2C, permitindo seu controle diretamente pelo ESP8266. Percebeu-se também a necessidade de um botão de saída, para destravar a porta pelo lado interno do ambiente.

Essa última necessidade, juntamente com a substituição do *display* LCD, permitiu que o hardware pudesse ser dividido em um módulo externo, contendo o ESP8266, leitor RFID, conversor lógico e interface com o usuário, e um módulo interno, com os demais componentes. Tal divisão agregou dois aspectos positivos ao terminal, decorrentes também do uso do ATmega328. O primeiro foi a resposta imediata ao acionamento do botão de saída, imprescindível em casos de emergência e em possíveis atrasos ou falhas no software do ESP8266. Já o segundo benefício foi relativo à segurança: uma vez que o atuador não é acionado diretamente por um simples nível lógico controlado pelo ESP8266, e sim por um comando enviado pelo barramento I2C, há uma maior dificuldade em destravar a porta por meio de uma manipulação indevida do módulo externo. Por exemplo, mesmo que se desconecte os fios do módulo externo a porta não se abrirá, a menos que seja enviado tal comando I2C. Esse fato aumenta bastante a segurança do sistema desenvolvido.

No Apêndice B pode-se ver o novo circuito elétrico do terminal, divido nos módulos interno e externo, desenvolvido a partir das melhorias concebidas. Já na Figura 16 estão expostos os módulos fabricados. A conexão entre ambos é feita utilizando um cabo de par trançado de 8 vias, utilizado em cabeamento Ethernet, com

conectores do tipo 8P8C, popularmente conhecidos como RJ45. A escolha desse tipo de conexão se deu devido à sua popularidade e facilidade de confecção.

Figura 16 – Protótipo final de hardware do terminal, dividido em módulos interno e externo.



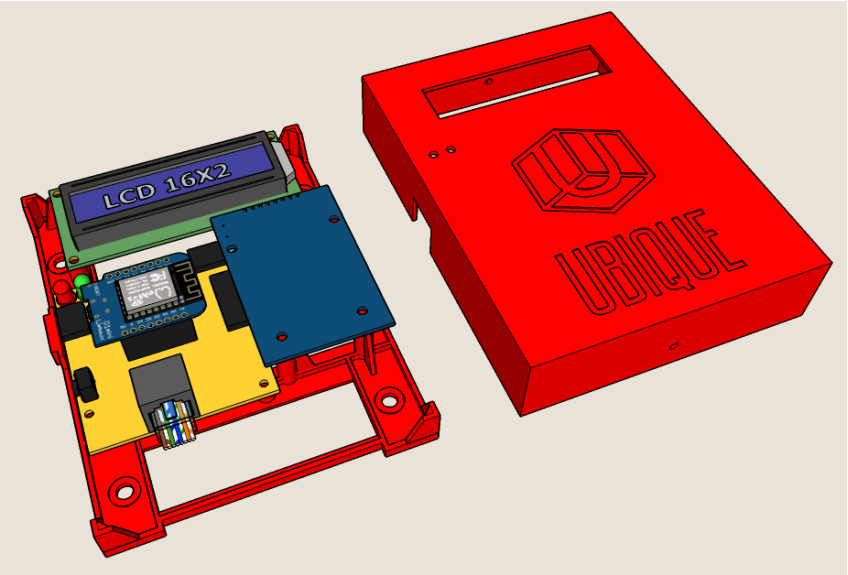
Fonte: Próprio autor.

Para o módulo externo, foi projetada e confeccionada uma caixa plástica personalizada, com suportes para a placa principal, *display* LCD e leitor RFID. Disponibilizou-se uma entrada na parte lateral esquerda para um conector de comunicação serial RS232. A modelagem 3D da caixa e dos componentes pode ser vista na Figura 17, e a caixa confeccionada pode ser vista na Figura 18. Para oferecer maior resistência à caixa foram confeccionadas também peças de aço que serviram como suporte para a tampa.

A ordem dos pinos do conector serial foi estabelecida de forma a facilitar a conexão de um módulo de comunicação Bluetooth HC-05, permitindo assim o acesso às configurações e informações de funcionamento do terminal por meio de um smartphone ou notebook, sem a necessidade de um cabo serial. Na Figura 19 é possível ver o módulo HC-05 conectado à porta serial e a visualização dos dados em um smartphone Android, por meio do aplicativo *Bluetooth Terminal HC-05*[1]. Tais dados foram transmitidos após a leitura de um cartão de um administrador, que habilitou temporariamente o fluxo de dados por meio da porta serial. É possível também enviar comandos para o terminal por meio do aplicativo.

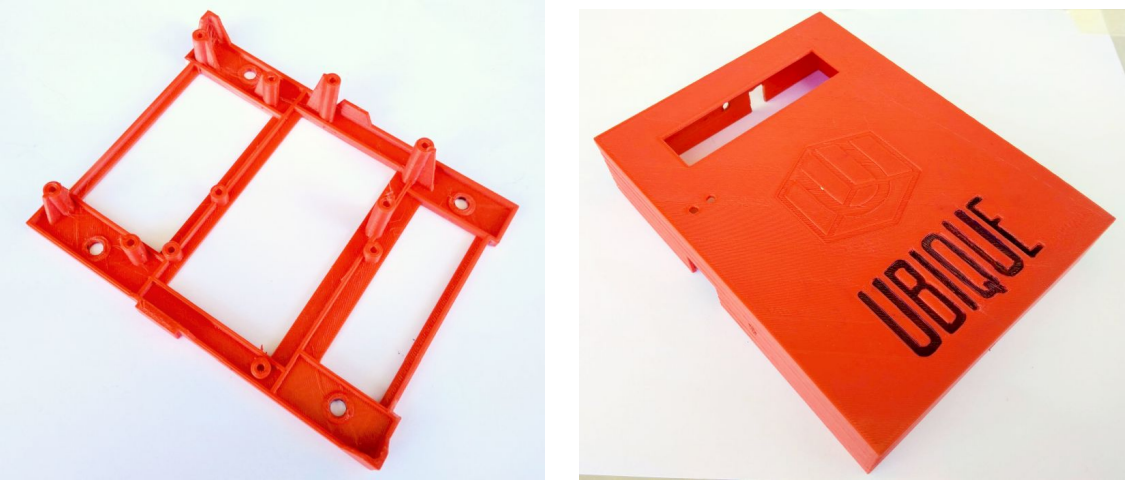
[1] Disponível em: <https://play.google.com/store/apps/details?id=project.bluetoothterminal>. Acesso em: 6 jun 2018.

Figura 17 – Modelagem 3D da caixa e dos componentes do módulo externo do terminal.



Fonte: Próprio autor.

Figura 18 – Partes da caixa externa produzidas por meio do método de impressão 3D.



(a) base (b) tampa

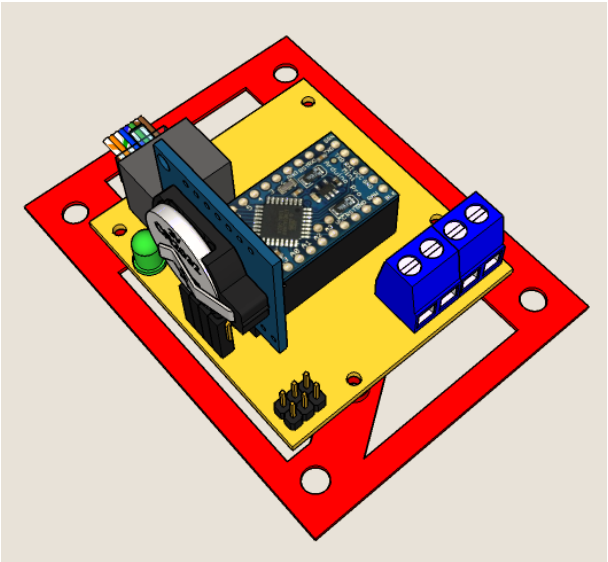
Fonte: Próprio autor.

Figura 19 – Demonstração do acesso à interface do administrador por meio da comunicação Bluetooth entre um smartphone e o módulo HC-05.

Fonte: Próprio autor.

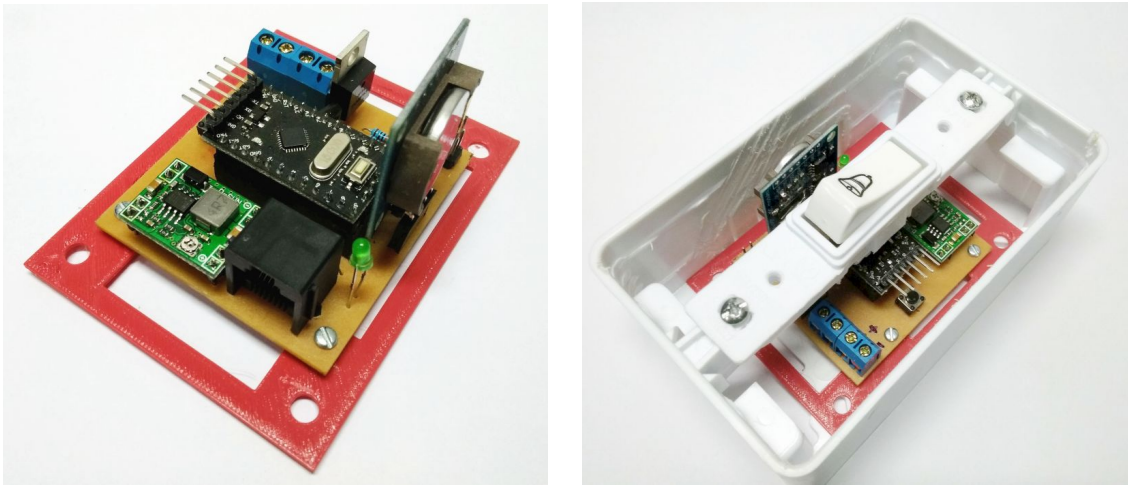
Para o módulo interno, utilizou-se uma caixa de passagem elétrica do modelo utilizado no padrão Sistema X, juntamente com um pulsador de campainha como botão de saída. Esse conjunto foi escolhido por ser robusto, de baixo custo e de fácil acesso no mercado. Para fixar a placa de circuito impresso do módulo interno utilizou-se uma base plástica, também projetada e confeccionada utilizando o método de impressão 3D. Na Figura 20 é exibida a modelagem dessa base e do módulo interno. Já na Figura 21 pode-se ver a base e as placas dispostas dentro da caixa de passagem.

Figura 20 – Modelagem 3D da base e dos componentes do módulo interno do terminal.



Fonte: Próprio autor.

Figura 21 – Partes do módulo interno.



(a) módulos montados na base (b) caixa de passagem e pulsador

Fonte: Próprio autor.

Para validar o sistema, instalou-se um terminal de controle de acesso no LASID, fazendo uso do módulo de alimentação de um sistema de acesso comercial instalado anteriormente. Tal sistema conta com um teclado numérico para identificação pessoal (como visto na Figura 22a), uma trava eletromagnética instalada na porta de correr de vidro, e um módulo de alimentação ininterrupta (como visto na Figura 22b) que mantém a trava energizada em caso de falta de energia.

Figura 22 – Sistema de controle de acesso comercial instalado no LASID.

(a) teclado numérico (b) fonte de alimentação ininterrupta

Fonte: Próprio autor.

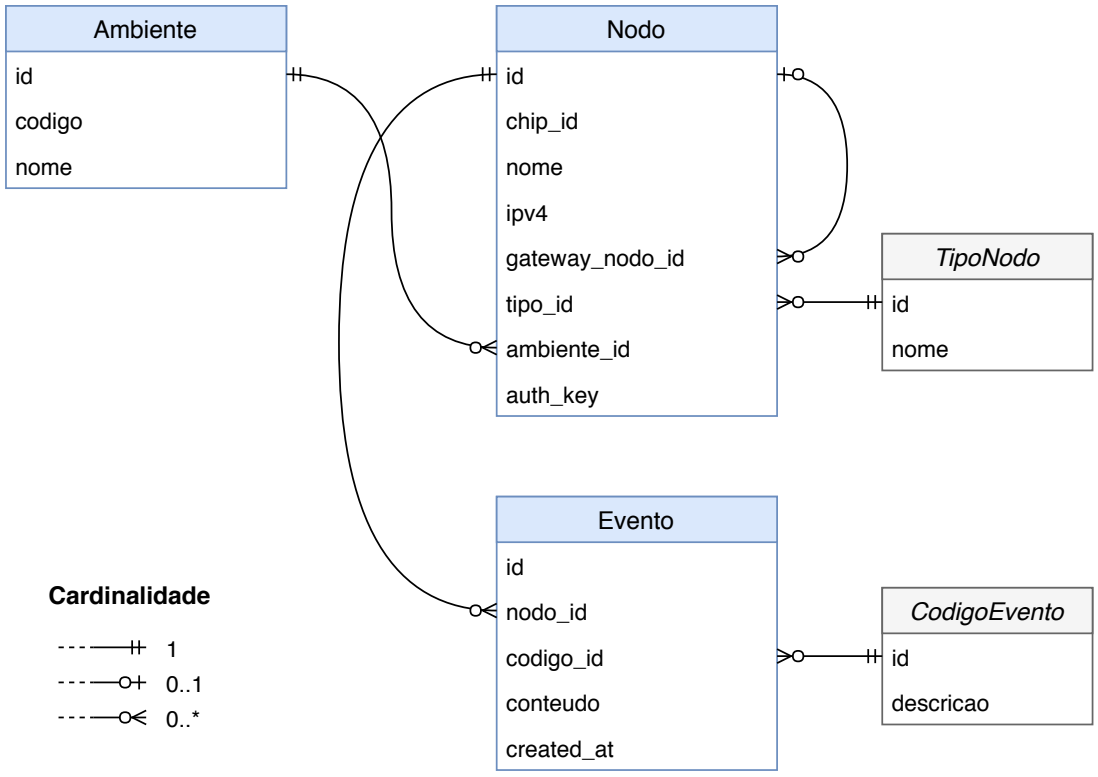
6 ARQUITETURA E VALIDAÇÃO

Esta seção apresenta as etapas percorridas no desenvolvimento da estrutura comum do Projeto Ubique, composta pelas principais entidades e pelo servidor web. São mostrados também os passos da construção do software do sistema de controle de acesso, configurações da rede e do servidor, e por fim, os testes e validações executados.

6.1 MODELAGEM DO SOFTWARE

O primeiro passo no desenvolvimento do software foi elencar as entidades do sistema. Na rede de sensores sem fios foram identificadas como entidades os Nodos e os Ambientes em que estão instalados. As informações coletadas pelos nodos são representadas por uma entidade genérica, Evento, que contém a data e hora do registro, um código que indica o tipo de evento, e a identificação do nodo que o originou. Tais entidades, seus atributos e relacionamentos podem ser vistos na Figura 23.

Figura 23 – Diagrama de entidades-relacionamentos do Projeto Ubique.



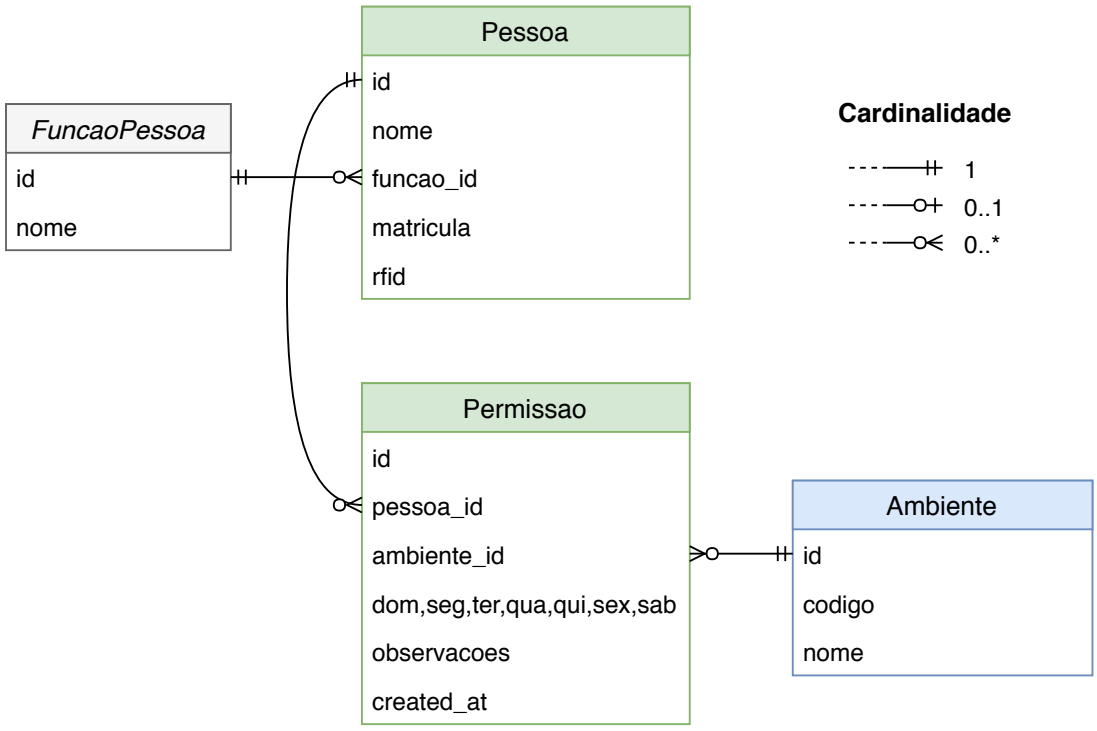
Fonte: Próprio autor.

As entidades TipoNodo e CodigoEvento foram concebidas para oferecer suporte aos diferentes sistemas que podem ser integrados ao Projeto Ubique. Cada sistema pode estabelecer os tipos de nodos e os códigos de eventos que forem necessários, bem como interpretar de maneiras distintas o campo conteudo de Evento, de acordo com o código associado.

Na entidade Nodo, o atributo gateway_nodo_id, quando presente, indica que o nodo em questão não é acessível diretamente pelo servidor, estando assim conectado a um nodo *gateway* como mostrado na Figura 3. Dessa forma, quando o servidor precisar enviar uma mensagem ao nodo fim, deverá utilizar o nodo *gateway* indicado como intermediário da comunicação. Previu-se também uma autenticação básica por meio do atributo auth_key, armazenado no nodo e no servidor, e que deve estar presente e ser válido nas mensagens trocadas entre ambos.

No sistema de controle de acesso, foram identificadas como entidades as Pessoas que terão acesso aos ambientes, por meio da concessão de Permissao para cada ambiente. Tais permissões são armazenadas em cada nodo de acesso, evitando assim consultas ao servidor para cada cartão lido e consequentemente diminuindo o tempo de resposta. A relação entre essas entidades pode ser vista na Figura 24.

Figura 24 – Diagrama de entidades-relacionamentos do sistema de controle de acesso.



Fonte: Próprio autor.

A entidade FuncaoPessoa é utilizada para diferenciar alunos, professores, funcionários terceirizados e administradores do sistema. Na entidade Permissao, cada

dia da semana é representado por um atributo do tipo binário.

Para ajudar na identificação das funcionalidades desejadas dos softwares do terminal e do servidor, foram levantados os requisitos funcionais e não-funcionais do sistema de controle de acesso como um todo. O resultado pode ser visto no Apêndice A.

6.2 COMUNICAÇÃO

O desenvolvimento do software do sistema de controle de acesso é dividido em dois subsistemas, sendo eles o terminal e o servidor web. A comunicação entre ambos se dá por meio do protocolo TCP/IP, com as informações codificadas em JSON (do inglês, *JavaScript Object Notation*). Como o servidor web é baseado no protocolo HTTP, os nodos precisam adicionar um cabeçalho desse protocolo às requisições para o servidor. Tal cabeçalho não se faz necessário para o caminho inverso, uma vez que o servidor é capaz de estabelecer uma conexão TCP/IP diretamente com os nodos. A seguir são apresentados exemplos de mensagens trocadas entre o servidor e os nodos.

Quando uma permissão de acesso a um ambiente é cadastrada no servidor, uma mensagem é enviada ao nodo de controle de acesso daquele ambiente. Tal mensagem contém o *id* do usuário no banco de dados, seu nome, o *id* do cartão inteligente, os dias em que o acesso é permitido e um parâmetro que indica se o usuário é administrador, tal como no Código 1. O campo `op` indica o cadastro de uma permissão no terminal, `auth_key` é a chave de autenticação do nodo, e o campo `dias` é interpretado como um número binário, onde cada bit corresponde à permissão de cada dia da semana.

Código 1 – Exemplo de mensagem JSON para cadastro de permissão.

```
{ "op"   : "i",
  "auth_key": "977813ef4074d0b708fb",
  "id"   : 7,
  "nome": "Gabriel Soares",
  "rfid": "3B1298DC",
  "dias": 126,
  "adm" : 1 }
```

Quando um cartão é lido pelo terminal de acesso, um evento é registrado contendo o *id* do cartão, o tipo de evento (acesso autorizado ou negado) e a data e hora em que ocorreu a leitura. Tais dados também são enviados ao servidor no formato JSON, porém com um cabeçalho HTTP, como no Código 2.

No cabeçalho, além dos parâmetros padrão do protocolo HTTP, são adicionadas também as informações de autenticação do *gateway*, a fim de validar a requisição.

Código 2 – Exemplo de mensagem de envio de permissão em JSON com cabeçalho HTTP.

```
POST /api/eventos HTTP/1.1
Host: 67.73.3.2
Accept: application/json
Content-Type: application/json
Content-Length: 219
X-gateway-chip-id: 14303114
X-gateway-auth-key: 977813ef4074d0b708fb

{ "chip_id" : 14303114,
  "auth_key": "977813ef4074d0b708fb",
  "codigo_mensagem": 1,
  "mensagem": {
      "codigo_evento": 101,
      "created_at"   : "2018-05-30T18:05:00",
      "conteudo"     : "3B1298DC" } }
```

Como no exemplo apresentado a mensagem foi enviada pelo próprio nodo *gateway*, tais dados de autenticação estão repetidos na mensagem. O campo `codigo_mensagem` indica o tipo de entidade cujas informações (`mensagem`) estão sendo enviadas. Nesse exemplo, o valor 1 corresponde a um Evento.

Além de indicar acessos autorizados e negados, o terminal utiliza os eventos para registrar também abertura da porta, inicialização do sistema e falhas internas de hardware e de comunicação.

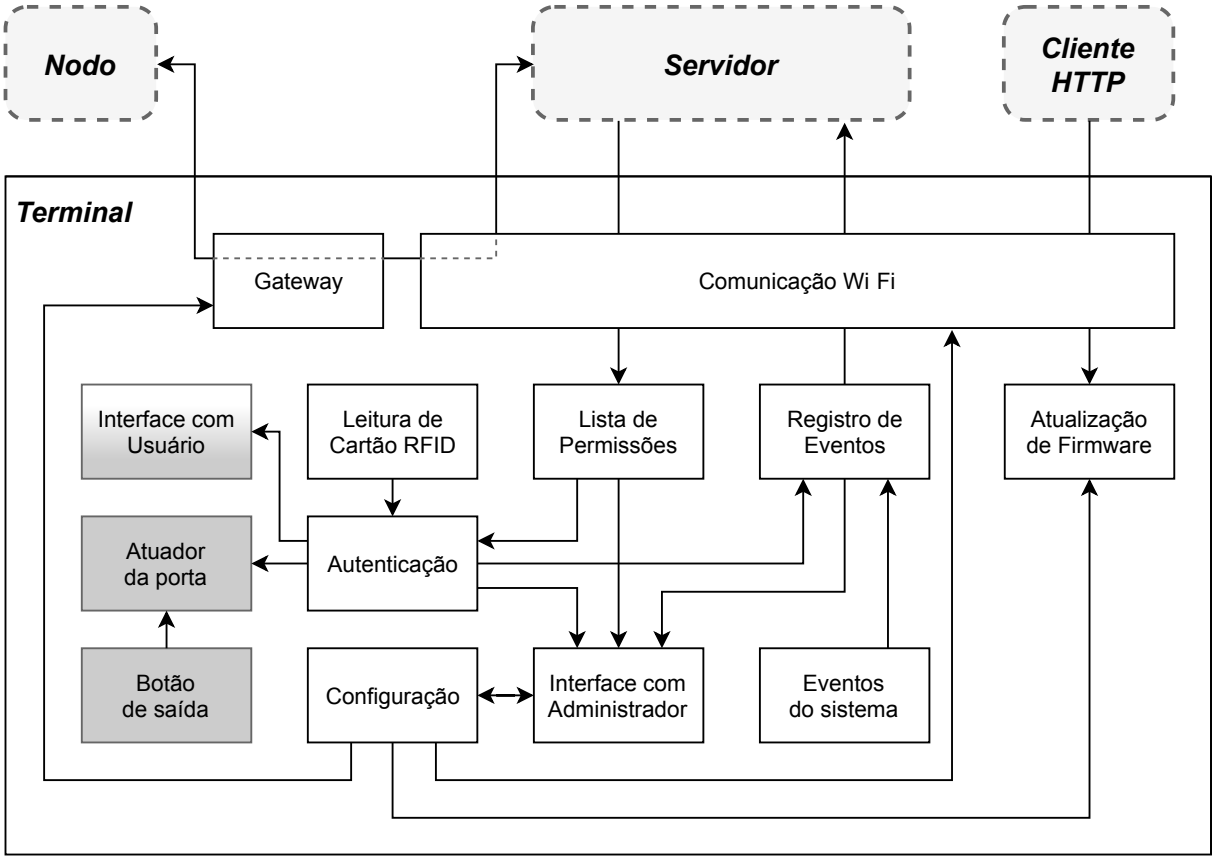
6.3 SOFTWARE DO TERMINAL DE ACESSO

O software do terminal de controle de acesso foi desenvolvido na linguagem C++, usando como base o *framework* Arduino e bibliotecas para interface com o leitor RFID, RTC, *display* LCD e para comunicação Wi-Fi. Foram identificadas as funcionalidades desejadas e suas relações, resultando no diagrama da Figura 25.

Mesmo dentro do terminal foram desenvolvidos dois softwares distintos (chamados *firmwares*), sendo o principal para o ESP8266 (blocos brancos no diagrama), que abrange as funções de controle de acesso, registro de eventos, comunicação com periféricos e com o servidor, e o secundário, mais simples, para o ATmega328 (funções em cinza no diagrama) controlar os periféricos (LEDs, *buzzer*, atuador da porta e botão de saída). A interface com o usuário, em gradiente na Figura 25, é controlada tanto pelo ESP8266 (*display* LCD) quanto pelo ATmega328 (LEDs e *buzzer*).

O servidor é responsável por inserir e remover permissões no terminal de acesso e por receber uma cópia dos eventos registrados pelo terminal. Tanto as permissões quanto os eventos são registrados na memória interna do terminal, e

Figura 25 – Diagrama de blocos do software do terminal de controle de acesso.



Fonte: Próprio autor.

podem ser consultados pelo administrador.

Além dessas funções, a interface com o administrador pode ser utilizada também para verificar e atualizar as configurações de rede e de hardware, bem como para verificar dados da rede e do sistema e para liberar a função de atualização do *firmware*. Tais funções são acessadas por meio da porta de comunicação serial RS232, disponibilizada pelo ESP8266.

O *framework* Arduino oferece, por padrão, uma classe que permite a leitura e escrita de dados na porta de comunicação serial. Como as funções de configuração são acessadas por meio dessa porta, foi necessário implementar uma classe adicional que controlasse o fluxo de dados. Dessa forma, todos os acessos à porta serial executados pelo software do terminal passam por essa classe de controle. Por padrão, a classe opera de forma bloqueada, isto é, nenhum dado é lido ou escrito na porta serial. O acesso é liberado temporariamente quando o terminal lê o cartão RFID de um administrador, que pode assim ter acesso às funções de configuração, podendo inclusive liberar de forma permanente o fluxo de dados.

Caso o terminal esteja configurado como *gateway*, o ESP8266 passa a operar

também como ponto de acesso, ao qual outros nodos podem se conectar e, assim, se comunicar com o servidor. A operação no modo *gateway* é diferente de acordo com a direção do fluxo dos dados, como detalhado a seguir.

Quando o nodo fim deseja se comunicar com o servidor, ele deve enviar a mensagem para o nodo *gateway*, instruindo-o para redirecioná-la para o servidor. No Código 3 é mostrado um exemplo de mensagem recebida pelo *gateway*, emitida por um de seus nodos fim. Ao recebê-la, o nodo *gateway* interpreta a operação como sendo de redirecionamento para o servidor, e em seguida extrai a mensagem do nodo fim, adiciona o cabeçalho HTTP e realiza o envio da mensagem mostrada no Código 4.

Código 3 – Exemplo de mensagem enviada ao nodo *gateway* para ser retransmitida ao servidor.

```
{ "op": "f",
  "mensagem"  : {
      "chip_id" : 13066734,
      "auth_key": "6c09ee1b9b120d4c6fbb",
      "codigo_mensagem": 1,
      "mensagem": {
          "codigo_evento": 101,
          "created_at"   : "2018-06-05T13:47:38",
          "conteudo"     : "E2C848AB" } } }
```

Código 4 – Exemplo de mensagem enviada ao servidor, retransmitida pelo nodo *gateway*.

```
POST /api/eventos HTTP/1.1
Host: 67.73.3.2
Accept: application/json
Content-Type: application/json
Content-Length: 219
X-gateway-chip-id: 14303114
X-gateway-auth-key: 977813ef4074d0b708fb

{ "chip_id" : 13066734,
  "auth_key": "6c09ee1b9b120d4c6fbb",
  "codigo_mensagem": 1,
  "mensagem": {
      "codigo_evento": 101,
      "created_at"   : "2018-06-05T13:47:38",
      "conteudo"     : "E2C848AB" } }
```

Quando o servidor deseja enviar uma mensagem para o nodo fim, deve instruir o nodo *gateway* a repassá-la para o nodo com o `chip_id` correspondente. O Código 5 mostra um exemplo de mensagem JSON com essa finalidade.

Para enviar o conteúdo da mensagem para o nodo fim, o nodo *gateway* realiza antes uma busca em todos os nodos conectados, procurando o endereço IP

Código 5 – Exemplo de mensagem enviada ao nodo *gateway*, a ser retransmitida para o nodo fim.

```
{ "op"      : "g",
  "chip_id" : 13066734,
  "mensagem": {
      "op"  : "i",
      "auth_key": "6c09ee1b9b120d4c6fbb",
      "id"  : 10,
      "nome": "Eudisley Gomes",
      "rfid": "2B66BE10",
      "dias": 126,
      "adm" : 1 } }
```

correspondente ao `chip_id` indicado. Tal busca se faz necessária porque o *gateway* atribui endereços IP dinâmicos aos seus nodos fim, e não mantém um registro dos nodos conectados e seus respectivos valores de `chip_id`. Entretanto, um algoritmo de registro temporário desses valores pode ser implementado posteriormente.

Para evitar a remoção do terminal do seu local de instalação, foi implementada uma função de atualização do *firmware* por meio da rede Wi-Fi. Quando a configuração é ativada, o terminal passa a responder requisições HTTP com uma página HTML para *upload* do arquivo binário do *firmware*, que pode ser acessada por qualquer cliente HTTP (um navegador web em um computador ou *smartphone*, por exemplo) conectado à rede Ubique. Uma vez feito o *upload*, o sistema é atualizado automaticamente e a configuração de atualização é desativada. Além do acesso a essa configuração estar protegido pela interface com o administrador, para acessar a página de *upload* é preciso fornecer um usuário e uma senha válidos, configurados na memória interna do terminal.

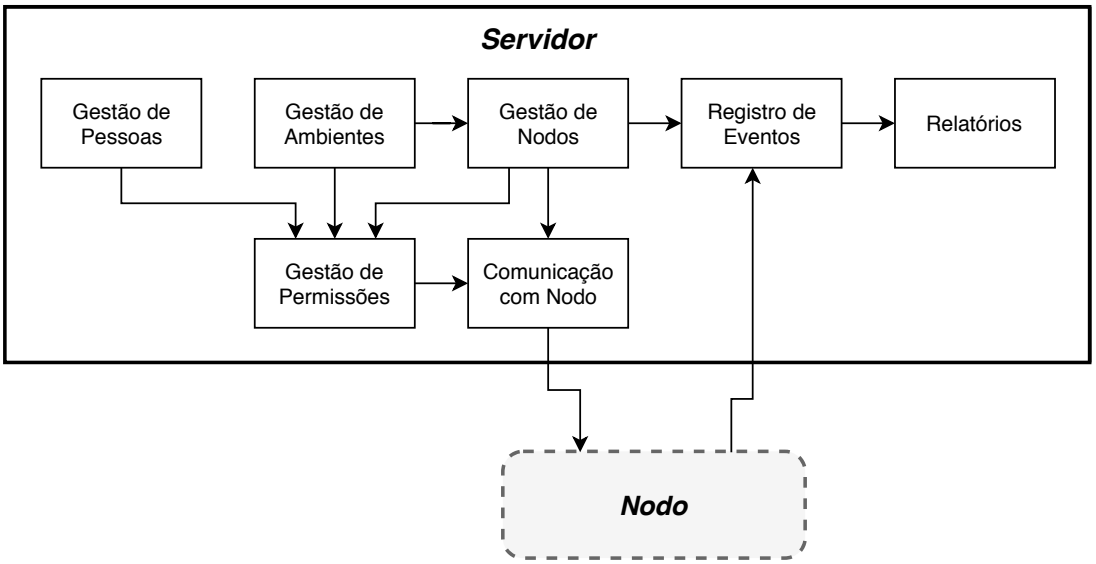
6.4 SOFTWARE DO SERVIDOR WEB

O servidor web foi desenvolvido na linguagem PHP, utilizando como base o *framework* Laravel. Para armazenar os dados, adotou-se um banco de dados MySQL. O desenvolvimento do sistema foi feito em um ambiente Microsoft Windows 7, utilizando o pacote Uniform Server[1] que agrupa os softwares Apache, PHP e MySQL.

Assim como no desenvolvimento do software do terminal, foram identificadas inicialmente as funções desejáveis e suas relações, que podem ser vistas na Figura 26. O termo "gestão" implica em funcionalidades de cadastro, consulta e atualização dos dados das entidades correspondentes. As relações com o Nodo são separadas devido à diferença no processo de comunicação, como explicado na seção 6.2.

[1] Disponível em: <http://www.uniformserver.com/>. Acesso em: 6 jun 2018.

Figura 26 – Diagrama de blocos do software do servidor web.



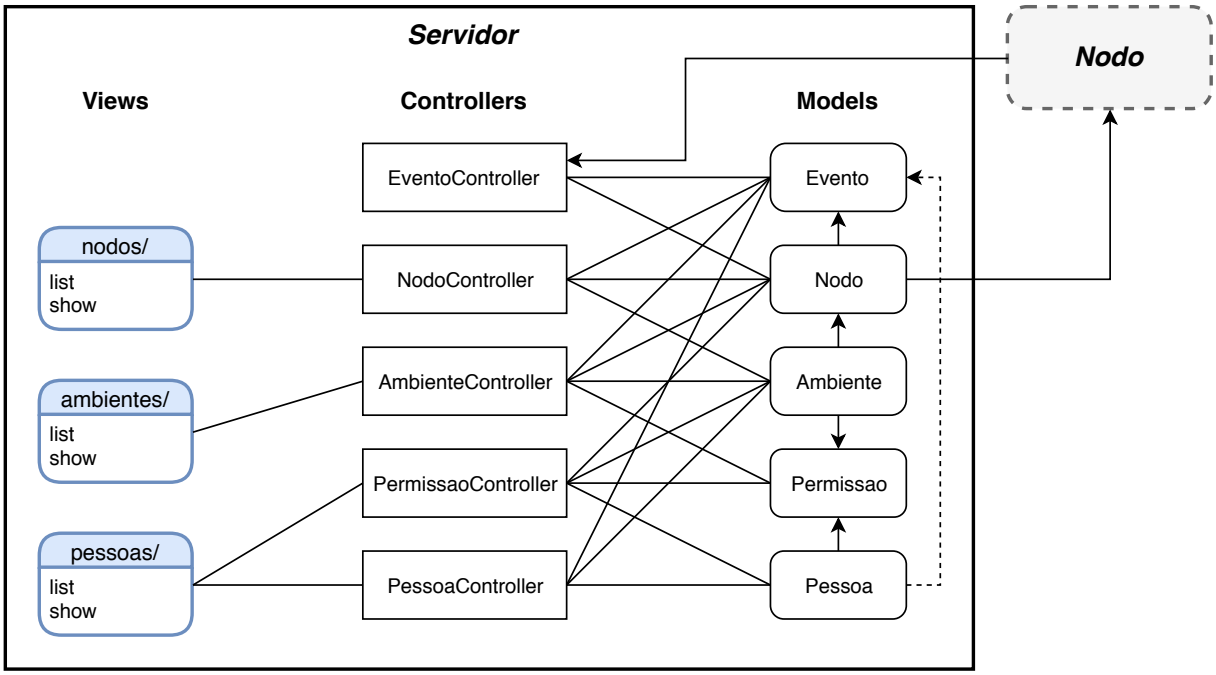
Fonte: Próprio autor.

As classes principais do software do servidor, que implementam o padrão de projeto MVC, e seus relacionamentos podem ser vistos na Figura 27. Cada item dentro dos grupos de *views* de cada entidade (`nodos`, `ambientes` e `pessoas`) representa uma página da interface. As *views* `list` apresentam a lista de entidades correspondentes e contém um formulário para cadastro e edição de dados. As *views* `show` mostram as informações detalhadas de cada entidade e as demais entidades relacionadas. Por exemplo, a página `nodos/show`, além de mostrar os detalhes de um nodo, exibe também os eventos registrados por ele. Já a página `pessoas/show` permite a visualização, o cadastro e a remoção das permissões de acesso da pessoa que está sendo mostrada.

Os *controllers* são os pontos de entrada do software do servidor. São responsáveis por receber as requisições HTTP, processá-las e retornar uma resposta, seja como uma página HTML da interface, requisitada por um navegador web, ou como uma simples mensagem de resposta ao registro de um evento enviado por um nodo, como é o caso da classe `EventoController`. As linhas entre os *models* e os *controllers* indicam relações de uso, ou seja, o *controller* consulta instâncias daquelas entidades, além de criar e atualizar instâncias do seu *model* correspondente.

As direções das setas entre os *models* indicam os relacionamentos entre as entidades, modeladas na seção 6.1. Por exemplo, uma instância da entidade `Permissao` armazena referências para o `Ambiente` ao qual está relacionada e para a `Pessoa` a que pertence. Da mesma forma, um `Nodo` armazena uma referência ao `Ambiente` em que está instalado. A entidade `Evento`, por sua vez, possui uma referência indireta à `Pessoa`, que existe apenas em eventos de leitura de cartão, por meio do número RFID registrado no campo `conteudo` (vide Figura 23). As entidades auxiliares `TipoNodo`,

Figura 27 – Diagrama das principais classes do servidor web.



Fonte: Próprio autor.

`CodigoEvento` e `FuncaoPessoa` foram omitidas do diagrama para simplificá-lo, e não possuem *controllers* associados.

A funcionalidade de relatórios indicada no diagrama da Figura 26 está distribuída nas páginas `show` de cada entidade. Por exemplo, a página `nodos/show` mostra todos os eventos registrados pelo nodo sendo exibido, enquanto a página `ambientes/show` mostra os eventos registrados pelos nodos localizados naquele ambiente.

No diagrama da Figura 27 é possível observar também a diferença no processo de comunicação do servidor com o nodo. O registro de eventos, por ser feito utilizando requisições HTTP, é controlado pela classe `EventoController`, que trata requisições desse tipo. Já o envio de dados ao nodo é feito diretamente pela classe `Nodo`, por meio de uma conexão TCP/IP estabelecida com o nodo desejado ou com o nodo *gateway* correspondente.

6.5 CONFIGURAÇÃO DA REDE E DO SERVIDOR

Para conectar os nodos da rede sem fios com o servidor, foi configurado um ponto de acesso exclusivo para o Projeto Ubique. Foi feita a alteração do login e senha da interface de administração do equipamento, e configurada uma rede Wi-Fi com autenticação WPA2 e senha de 24 caracteres aleatórios, contendo letras maiúsculas, minúsculas e números. Tais características invalidam tentativas de quebra da senha

utilizando ataques baseados em dicionários (NULL BYTE, 2017).

O recurso de autenticação WPS do ponto de acesso foi desativado por não ser necessário, e também por apresentar vulnerabilidades de segurança (PASH, 2012). O serviço de DHCP, responsável por atribuir endereços IP aos clientes da rede, também foi desativado com o intuito de reduzir o tempo de estabelecimento da conexão dos nodos à rede. Por esse motivo, configurou-se em cada nodo da rede um IP estático, que não sofre alterações ao longo do tempo.

Após o desenvolvimento do software do servidor web em um computador de testes (descrito na seção 6.4), configurou-se um computador servidor para disponibilizar o sistema para os nodos da rede. Tal computador contava com o Linux Mint 18 Cinnamon, no qual foram instalados os softwares Apache, PHP e MySQL. Foram configuradas senhas fortes para o banco de dados MySQL, compostas de 24 caracteres aleatórios contendo letras maiúsculas, minúsculas, números e símbolos.

Em seguida, algumas configurações do sistema operacional foram alteradas para aumentar a segurança do servidor (KENNEDY, 2013). Para o gerenciamento remoto, configurou-se o método de autenticação por meio de chave pública, diferente do método padrão que requer apenas o nome de usuário e senha. No método de chave pública, o usuário deve gerar um par de chaves criptográficas (uma pública e uma privada), armazenar a chave pública no servidor, e utilizar a chave privada para autenticar-se no servidor e assim executar comandos remotamente utilizando o protocolo SSH (SSH INC, 2017).

O software Apache foi configurado para ocultar a exibição da versão instalada (evitando a exploração de vulnerabilidades específicas dessa versão), impedir a listagem de arquivos de diretórios do sistema e foram também desabilitados módulos desnecessários (SHRIVASTAVA, 2016). O acesso ao sistema web do Ubique foi restringido apenas para os endereços IP da rede Ubique, por meio da configuração do arquivo `/etc/apache2/sites-available/ubique.conf` como mostrado no Código 6.

Para que o computador do servidor pudesse ter acesso à rede de sensores, instalou-se um adaptador de rede USB além da interface de rede *onboard* já existente. A existência do adaptador adicional fez com que o sistema operacional tentasse utilizá-lo como padrão para acessar a Internet, ignorando a interface *onboard* que estava conectada de fato à rede externa. Dessa forma, foram necessárias configurações adicionais para instruir o sistema operacional a utilizar o adaptador USB apenas para o tráfego da rede Ubique (UBUNTU GEEK, 2008). Tal configuração foi executada ao editar o arquivo `/etc/network/interfaces` e incluir o Código 7, que estabelece a configuração manual da rede Ubique e remove as configurações automáticas atribuídas ao adaptador USB. O código `enx00e04c534458` é a identificação do adaptador de rede dada pelo sistema operacional, que foi obtido executando o

Código 6 – Arquivo de configuração do software Apache para o servidor web.

```
# Arquivo /etc/apache2/sites-available/ubique.conf
<Directory "/var/www/ubique/public">
    Allowoverride All
    Require all granted
    Options +Indexes +FollowSymLinks +MultiViews
</Directory>

<VirtualHost 67.73.3.2:80 *:80>
    ServerName ubique.lumo
    ServerAdmin gutierrez@eng.ci.ufpb.br
    DocumentRoot /var/www/ubique/public

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

comando `ip route list`[2]. Os endereços IPs `67.73.3.1` e `67.73.3.2` correspondem, respectivamente, aos endereços do ponto de acesso e do servidor web.

Código 7 – Arquivo de configuração do adaptador de rede USB para conexão com a rede Ubique.

```
# Arquivo /etc/network/interfaces
auto enx00e04c534458
iface enx00e04c534458 inet static
    address 67.73.3.2
    netmask 255.255.255.0
    gateway 67.73.3.1

up ip route del default via 67.73.3.1
```

6.6 TESTES E VALIDAÇÃO

Cada etapa do desenvolvimento do sistema de controle de acesso contou com testes específicos. Desenvolveu-se primeiramente o servidor, uma vez que ele é o responsável por adicionar e remover permissões do terminal, e por registrar os eventos. Inicialmente, foram testadas as funcionalidades de cadastro, listagem, atualização e remoção das instâncias das entidades `Nodo`, `Ambiente`, `Pessoa` e `Permissao` no banco de dados.

O passo seguinte consistiu no desenvolvimento do terminal em duas partes. Na primeira etapa, os componentes de hardware foram sendo testados à medida que iam sendo conectados para formar o circuito elétrico final. Após validado o

[2] Disponível em: <https://serverfault.com/a/836708>. Acesso em: 25 abr 2018.

circuito, foram confeccionadas as placas de circuito impresso, e executados novamente os testes de integração e comunicação entre os diferentes módulos físicos. Em seguida, desenvolveu-se a parte de comunicação com o servidor, que compreende o armazenamento de permissões na memória local, bem como o registro de eventos no servidor e também na memória local. Foram feitos testes de inserção e remoção de permissões, incluindo a leitura de diferentes cartões RFID antes e após o cadastro da permissão no terminal, assim como a leitura após a remoção da permissão. Ao mesmo tempo testou-se também o registro das leituras dos cartões na memória local, e o envio dos eventos para o servidor.

Uma vez validadas as funções de comunicação entre o servidor e o terminal, foram implementadas e validadas as funções de segurança. Primeiramente, adicionou-se o parâmetro `auth_key` para autenticação das mensagens. Foram testados o envio e a recepção de mensagens com esse parâmetro incorreto, além de testes com os demais parâmetros das mensagens ausentes ou incorretos. Depois foi implementada a interface do administrador no terminal e testado o controle do fluxo de dados na porta serial ao apresentar um cartão de administrador, e em seguida outros cartões com e sem permissão de acesso ao ambiente.

Para validar o Projeto Ubique e o sistema de controle de acesso, instalou-se um terminal no LASID e foram distribuídos cartões RFID para os membros do laboratório. Durante esse processo, verificou-se que o crachá dos docentes e técnicos administrativos da UFPB era também um cartão inteligente, compatível com a versão RFID de 13,56 MHz adotada no sistema. Tal fato permitiu a associação desses crachás aos cadastros da pessoas no sistema, que assim puderam utilizá-los para liberar a entrada nos ambientes.

Por meio da instalação do terminal e distribuição dos cartões, foi possível testar o funcionamento contínuo do hardware e software do terminal de acesso, e também o registro de eventos no servidor web. Até o presente momento não foram relatados problemas no uso do terminal pelos membros do laboratório.

Em testes posteriores de cadastro de permissões por meio do servidor, verificou-se que um cartão havia sido cadastrado com sucesso no terminal, porém ao utilizar o cartão no terminal o acesso foi negado. Ao se tentar remover a permissão por meio do servidor, o terminal indicava que a permissão não estava cadastrada na memória interna. Foi feita uma nova tentativa de cadastro de permissão, porém o problema persistiu. Ao se apresentar um cartão que já estava cadastrado anteriormente, o acesso também foi negado, o que indicava uma possível corrupção do registro interno de permissões. Tal hipótese confirmou-se ao efetuar a leitura das permissões através da interface com o administrador, cujo fluxo de dados teve que ser desativado por meio da gravação de um novo *firmware*, visto que o cartão de administrador não foi

reconhecido.

Tal problema pode ter sido causado por uma remoção de permissão executada antes do cadastro do novo cartão, devido ao método de registro de permissões na memória interna que estava em uso anteriormente. Nesse método, todas as permissões eram armazenadas em um único arquivo binário, e o processo de remoção de uma delas consistia em criar uma nova cópia desse arquivo binário sem a permissão a ser removida. Assim, é possível que os dados das permissões existentes tenham sido corrompidos durante a cópia para o novo arquivo. Foi desenvolvido então um outro método de registro baseado em um sistema de acesso similar[3], no qual as permissões são gravadas em arquivos individuais. Dessa forma, a inserção ou remoção de uma permissão consiste apenas em criar ou remover o arquivo correspondente, sem interferir nos demais arquivos. Contudo, testes posteriores devem ser executados para verificar o número máximo de permissões que pode ser armazenado na memória interna.

[3] Disponível em: <https://github.com/omersiar/esp-rfid>. Acesso em: 8 jun 2018.

7 CONCLUSÕES E TRABALHOS FUTUROS

A partir das demandas do Centro de Informática, identificadas em termos de tecnologias inteligentes que pudessem ser desenvolvidas, foi concebida, implementada e validada uma rede de sensores sem fios capaz de atender a essas demandas: o Projeto Ubique. Como forma de validação da rede de sensores, foi projetado, desenvolvido e validado um sistema de controle de acessos para ambientes de uso comum. Além de todo o projeto ter sido baseado em softwares *open source*, foi feito uso de componentes de fácil acesso no mercado nacional, facilitando assim a replicação posterior do software e hardware descrito neste trabalho.

Ao longo do desenvolvimento do Projeto Ubique, adotaram-se algumas medidas de segurança básicas no servidor web e na rede sem fios, como configurações de login e senhas mais seguras, além da desativação de recursos desnecessários que pudessem criar vulnerabilidades. Adotou-se também um mecanismo básico de autenticação das mensagens trocadas entre o servidor e o nodos. Porém, o tráfego de informações entre os dispositivos do projeto está protegido apenas pelos protocolos de segurança nativos da rede Wi-Fi. Um invasor que consiga se conectar à rede sem fios do projeto pode não só ter acesso às mensagens trocadas entre os nodos, como também enviar mensagens falsas, seja de cadastro de permissões ou de registro de eventos. Por isso, a segurança é uma importante questão que deve ser analisada tanto no escopo deste projeto como no âmbito das redes de sensores sem fios.

Segundo Jing et al. (2014), Nawir et al. (2016) e Hossain, Hasan e Skjellum (2017), a segurança da informação na Internet das Coisas é um dos grandes desafios atualmente devido às suas características e limitações. Uma vez que a tecnologia está permeando cada vez mais a sociedade, a segurança da informação é vital para preservar a privacidade dos indivíduos e garantir o bom funcionamento dos diversos sistemas interconectados. Roman, Najera e Lopez (2011) indicam que a criptografia é indispensável para a proteção de uma rede de dispositivos, porém tais algoritmos ainda estão sendo estudados e adaptados para as limitações dos sistemas embarcados. Como trabalho futuro, pode ser feito o levantamento de algoritmos compatíveis com o microcontrolador utilizado neste projeto, assim como comparativos e testes para assegurar que o desempenho dos sistemas inteligentes não seja afetado de forma significativa.

Além da área de segurança, podem ser desenvolvidos trabalhos futuros voltados ao aumento da robustez do sistema de controle de acesso e à testes de estresse da

rede de sensores sem fios, por meio do envio de diversas mensagens em sequência, originadas de diversos nodos. Além disso, pode-se utilizar a estrutura do Ubique para testar diferentes algoritmos de roteamento, além da versão básica implementada neste trabalho. Por fim, o Projeto Ubique foi desenvolvido de forma que novas soluções de inteligência para edificações possam também ser concebidas e integradas ao projeto, aumentando sua capacidade e abrangência.

Toda a produção descrita neste trabalho não teria sido concretizada sem o conteúdo visto em diversas disciplinas do curso de Engenharia de Computação. Foram aplicados diversos conceitos como Especificação de Requisitos de Software, Engenharia de Software, Bancos de Dados, Computação Pervasiva, Redes de Sensores sem Fio, Sistemas Embarcados e outros. Por meio deste projeto foi possível ter uma vivência prática de todo o conhecimento adquirido, indispensáveis para a solidificação do aprendizado e crescimento acadêmico e profissional.

REFERÊNCIAS

ARDUINO. *Introduction*. 2018. Disponível em: <https://www.arduino.cc/en/Guide/Introduction>. Acesso em: 4 jun 2018. Citado na página 29.

BROWN, E. *Web Development with Node and Express: Leveraging the JavaScript Stack*. O'Reilly Media, 2014. ISBN 9781491902288. Disponível em: <https://www.safaribooksonline.com/library/view/web-development-with/9781491902288/ch14.html>. Acesso em: 19 jun 2018. Citado na página 30.

BUCKMAN, A. H.; MAYFIELD, M.; BECK, S. B. M. What is a Smart Building? *Smart and Sustainable Built Environment*, v. 3, n. 2, p. 92–109, set. 2014. ISSN 2046-6099. Disponível em: <https://www.emeraldinsight.com/doi/full/10.1108/SASBE-01-2014-0003>. Citado 3 vezes nas páginas 12, 15 e 16.

CABA. *Bright Green Buildings: Convergence of Green and Intelligent Buildings*. Ottawa, 2008. 1–220 p. Disponível em: <http://www.caba.org/brightgreen>. Citado na página 15.

GREEN, H. L. High-Performance Buildings. *Innovations: Technology, Governance, Globalization*, v. 4, n. 4, p. 235–239, out. 2009. ISSN 1558-2477, 1558-2485. Disponível em: <http://www.mitpressjournals.org/doi/10.1162/itgg.2009.4.4.235>. Citado 2 vezes nas páginas 12 e 17.

HOSSAIN, M.; HASAN, R.; SKJELLUM, A. Securing the Internet of Things: A Meta-Study of Challenges, Approaches, and Open Problems. In: *2017 IEEE 37th International Conference on Distributed Computing Systems Workshops (ICDCSW)*. [S.l.: s.n.], 2017. p. 220–225. Citado 2 vezes nas páginas 18 e 50.

JING, Q. et al. Security of the Internet of Things: perspectives and challenges. *Wireless Networks*, v. 20, n. 8, p. 2481–2501, nov. 2014. ISSN 1022-0038, 1572-8196. Disponível em: <https://link.springer.com/article/10.1007/s11276-014-0761-7>. Citado na página 50.

JOHNSON CONTROLS. *High Performance Buildings Through Knowledge Based Integration*. Milwaukee, EUA, 2003. Citado na página 17.

KENNEDY, B. *My First 5 Minutes On A Server*. 2013. Disponível em: <https://plusbryan.com/my-first-5-minutes-on-a-server-or-essential-security-for-linux-servers>. Acesso em: 24 abr 2018. Citado na página 46.

KIRKLAND, D. *Fingerprints are Usernames, not Passwords*. 2013. Disponível em: <http://blog.dustinkirkland.com/2013/10/fingerprints-are-user-names-not.html>. Acesso em: 1 jun 2018. Citado na página 23.

KOPETZ, H. Internet of Things. In: *Real-Time Systems*. Boston, MA, EUA: Springer, 2011, (Real-Time Systems Series). p. 307–323. ISBN 978-1-4419-8236-0 978-1-4419-8237-7. Disponível em: <https://link.springer.com/chapter/10.1007/978-1-4419-8237-7_13>. Citado na página 18.

LARAVEL. *Laravel: A PHP framework for web artisans*. 2018. Disponível em: <https://github.com/laravel/laravel>. Acesso em: 4 jun 2018. Citado na página 30.

LPKF LASER & ELECTRONICS AG. *Manual ProtoMat S42*. 2.0. ed. Garbsen, Alemanha, 2006. Citado na página 27.

MENDEZ, D. M.; PAPAPANAGIOTOU, I.; YANG, B. Internet of Things: Survey on Security and Privacy. *Information Security Journal: A Global Perspective*, v. 27, n. 3, p. 162–182, maio 2018. ISSN 1939-3555, 1939-3547. ArXiv: 1707.01879. Disponível em: <http://arxiv.org/abs/1707.01879>. Citado na página 18.

NAWIR, M. et al. Internet of Things (IoT): Taxonomy of security attacks. In: *2016 3rd International Conference on Electronic Design (ICED)*. Phuket, Thailand: IEEE, 2016. p. 321–326. Citado na página 50.

NULL BYTE. *Cracking WPA2-PSK Passwords Using Aircrack-Ng*. 2017. Disponível em: <https://null-byte.wonderhowto.com/how-to/hack-wi-fi-cracking-wpa2-psk-passwords-using-aircrack-ng-0148366/>. Acesso em: 7 jun 2018. Citado na página 46.

O'GORMAN, L. Comparing passwords, tokens, and biometrics for user authentication. *Proceedings of the IEEE*, IEEE, v. 91, n. 12, p. 2021–2040, 2003. Citado na página 23.

PASH, A. *How to Crack a Wi-Fi Network's WPA Password with Reaver*. 2012. Disponível em: <https://lifehacker.com/5873407/how-to-crack-a-wi-fi-networks-wpa-password-with-reaver>. Acesso em: 24 abr 2018. Citado na página 46.

PLATFORMIO. *PlatformIO: an open source ecosystem for IoT development*. 2018. Disponível em: <https://platformio.org/>. Acesso em: 4 jun 2018. Citado na página 29.

ROMAN, R.; NAJERA, P.; LOPEZ, J. Securing the Internet of Things. *Computer*, v. 44, n. 9, p. 51–58, set. 2011. ISSN 0018-9162. Citado na página 50.

SHABHA, G. A critical review of the impact of embedded smart sensors on productivity in the workplace. *Facilities*, v. 24, n. 13/14, p. 538–549, nov. 2006. ISSN 0263-2772. Disponível em: <https://www.emeraldinsight.com/doi/abs/10.1108/02632770610705301>. Citado na página 15.

SHRIVASTAVA, T. *13 Apache Web Server Security and Hardening Tips*. 2016. Disponível em: <https://www.tecmint.com/apache-security-tips/>. Acesso em: 24 abr 2018. Citado na página 46.

SINOPOLI, J. *Smart Buildings Systems for Architects, Owners, and Builders*. Burlington, MA, EUA: Butterworth-Heinemann, 2010. ISBN 978-1-85617-653-8. Citado 2 vezes nas páginas 15 e 16.

SNOONIAN, D. Smart buildings. *IEEE Spectrum*, v. 40, n. 8, p. 18–23, ago. 2003. ISSN 0018-9235. Citado na página 17.

SOMMERVILLE, I. *Software Engineering*. 9. ed. Nova Iorque, EUA: Pearson, 2010. Citado 2 vezes nas páginas 29 e 30.

SSH COMMUNICATIONS SECURITY, INC. *Public Key Authentication for SSH*. 2017. Disponível em: <https://www.ssh.com/ssh/public-key-authentication>. Acesso em: 7 jun 2018. Citado na página 46.

UBUNTU GEEK. *Howto add permanent static routes in Ubuntu*. 2008. Disponível em: <http://www.ubuntugeek.com/howto-add-permanent-static-routes-in-ubuntu.html>. Acesso em: 25 abr 2018. Citado na página 46.

XDA DEVELOPERS. *Fingerprint Authentication – Just a Plain Bad Idea*. 2015. Disponível em: <https://www.xda-developers.com/fingerprint-authentication-just-a-plain-bad-idea/>. Acesso em: 1 jun 2018. Citado na página 23.

APÊNDICE A – DOCUMENTO DE REQUISITOS DO SISTEMA DE CONTROLE DE ACESSO

REQUISITOS FUNCIONAIS (RF)

[RF-01] GERENCIAMENTO DO SISTEMA

O gerenciamento das informações do sistema deve ser feito apenas por operadores habilitados.
Prioridade: essencial

[RF-02] GERENCIAMENTO DE USUÁRIOS

O sistema deve permitir o cadastro, consulta e exclusão de usuários candidatos para obter acesso, bem como a atualização de seus dados.
Prioridade: essencial

[RF-03] GERENCIAMENTO DE PERMISSÕES

O sistema deve permitir a concessão e revogação de permissões de acesso aos usuários cadastrados, respeitando as diretrizes do Centro.
Prioridade: essencial

[RF-04] GERENCIAMENTO DE TERMINAIS

O sistema deve permitir o cadastro, consulta, alteração e exclusão de terminais habilitados, que devem implementar as permissões de acesso cadastradas para o ambiente no qual está instalado.
Prioridade: importante

[RF-05] CONTROLE DE ACESSO

O sistema não deve permitir o acesso de pessoas não autorizadas aos ambientes gerenciados.
Prioridade: essencial

[RF-06] REGISTRO DE ACESSOS

O sistema deve armazenar concessões e negações de acesso aos ambientes cadastrados.
Prioridade: essencial

[RF-07] RELATÓRIOS DE ACESSOS

O sistema deve ser capaz de emitir, quando solicitado, relatórios de acessos concedidos e negados por ambiente e por usuário, em um determinado período de tempo.
Prioridade: importante

REQUISITOS NÃO-FUNCIONAIS (RNF)

Os requisitos não-funcionais foram classificados nos seguintes tipos: [CONF] confiabilidade, [SEG] segurança, [INTERF] interface, [USA] usabilidade e [DES] desempenho.

[RNF/CONF-01] TOLERÂNCIA À PERDA DE COMUNICAÇÃO

Os terminais devem ser capazes de validar credenciais mesmo quando não houver comunicação com o servidor, baseando-se nas permissões estabelecidas antes da interrupção, armazenadas em uma memória local.
Prioridade: essencial

[RNF/CONF-02] TOLERÂNCIA À FALTA DE ENERGIA

Em caso de falta de energia, o acesso aos ambientes deve continuar sendo restrito. O terminal pode ser alimentado por uma bateria, ou pode deixar de exercer o controle desde que a porta utilize um mecanismo de trava *fail-secure*, isto é, que mantenha a porta travada mesmo sem energia.
Prioridade: essencial

[RNF/SEG-03] COMUNICAÇÕES CRIPTOGRAFADAS

A comunicação entre os terminais e o servidor deve ser criptografada, de forma que terceiros não sejam capazes de decodificar e interpretar os dados trocados entre esses módulos.
Prioridade: importante

[RNF/INTERF-04] COMUNICAÇÃO SEM FIOS

A comunicação entre os terminais e os pontos mediadores da conexão com o servidor deve ser feita através de protocolo sem fios, de forma que o único barramento físico conectado aos terminais de acesso seja o de alimentação (e eventuais cabos entre módulos).
Prioridade: essencial

[RNF/INTERF-05] INTERFACE COM O USUÁRIO

O terminal deve ser dotado de interface que informe ao usuário se a credencial foi aceita ou não.
Prioridade: essencial

[RNF/INTERF-06] INTERFACE PARA ADMINISTRAÇÃO

O terminal deve possuir interface para visualização e edição das suas configurações. Tal interface só deve ser disponibilizada após a apresentação de credencial de administração.
Prioridade: desejável

[RNF/USA-07] FUNCIONAMENTO MECÂNICO PARALELO

O acesso ao ambiente deve ser possibilitado através de chave, caso o tipo de porta seja compatível. A porta deve também poder ser aberta por dentro, independente de credencial.
Prioridade: essencial

[RNF/DES-08] TEMPO DE RESPOSTA

Após apresentação da credencial, o sistema deverá informar o resultado da validação em até 5 segundos.
Prioridade: importante

APÊNDICE B – CIRCUITO ELÉTRICO DO TERMINAL DE CONTROLE DE ACESSO