UNIVERSIDADE VIRTUAL DO ESTADO DE SÃO PAULO

Jeferson Shiguemi Mukuno
Julio César Lisboa Coelho
Michael Jones da Silva
Rodrigo Nunes Sampaio Ribeiro
Tiago Garcia Barbedo

Projeto e implementação de um protótipo dispensador automático de ração para animais baseado em raspberry pi controlado por aplicativo móvel.

Link do video:
https://www.youtube.com/watch?v=_5dPdv_I79A

Teodoro Sampaio - SP
2021

UNIVERSIDADE VIRTUAL DO ESTADO DE SÃO PAULO

Projeto e implementação de um protótipo dispensador automático de ração para animais baseado em raspberry pi controlado por aplicativo móvel.

Relatório Técnico - Científico apresentado na disciplina de Projeto Integrador para o curso de Engenharia de Computação da Universidade Virtual do Estado de São Paulo (UNIVESP)

Teodoro Sampaio - SP
2021

MUKUNO, Jeferson Shiguemi; COELHO, Julio César Lisboa; SILVA, Michael Jones da; RIBEIRO, Rodrigo Nunes Sampaio; BARBEDO, Tiago Garcia. **Projeto e implementação de um protótipo dispensador automático de ração para animais baseado em raspberry pi controlado por aplicativo móvel.** 22f. Relatório Técnico-Científico. Engenharia de Computação – **Universidade Virtual do Estado de São Paulo**. Tutor: Everton Simões da Motta. Polo Teodoro Sampaio , 2021.

RESUMO

Este trabalho apresenta a implementação de um protótipo de um dispensador automático de ração para animais de estimação utilizando uma placa Raspberry Pi, controlado através de um software embarcado escrito em linguagem de programação python, utilizando as bibliotecas existentes de GPIO para controle dos periféricos utilizados. Utilizando um motor conectado a uma mola para empurrar a ração para fora do recipiente. A estrutura do protótipo foi construída em material plástico do tipo PVC para possibilitar uma fácil manutenção e limpeza. Através deste protótipo é possível determinar os horários para liberação do alimento e ainda é possível através de um aplicativo móvel acompanhar e controlar os históricos de alimentação de forma remota.

PALAVRAS-CHAVE: internet das coisas; alimentador automático; raspberry pi.

SUMÁRIO

1. INTRODUÇÃO

A Internet começou como uma rede que interligava duas universidades americanas e que foi agregando outras redes com o passar do tempo, atingindo o alcance global em alguns anos. O desenvolvimento da rede pode ser descrita em fases distintas: a primeira fase, onde era utilizada para conectar mainframes (computadores de grande porte), sendo uma rede de computadores; a segunda fase, onde se desenvolveu o contato do usuário final com a rede, com o uso de computadores pessoais e dispositivos portáteis, como notebooks e smartphones, e onde a Internet se tornou uma rede de pessoas e comunidades; e, finalmente, a terceira fase, que estamos vivenciando o seu início, é o da computação ubíqua, caracterizada pela presença constante e direta da informática e da tecnologia na vida das pessoas, em suas residências e nos ambientes de convívio social, principalmente nos ambientes profissionais.

Com a pandemia do COVID-19, uma grande parte dos trabalhadores passou a atuar em seus empregos de forma remota, atuando de suas casas por meio de sistemas de comunicação de áudio e vídeo, além de acessar os sistemas de informação das empresas, sempre utilizando-se da Internet como via de transmissão de dados.

Com esse esquema de trabalho, para o indivíduo, criou-se uma rotina estressante, onde tarefas domésticas se confundem, em um mesmo espaço físico, com atribuições profissionais, ocasionando problemas diversos, como, por exemplo, acúmulo de tarefas e desorganização.

No sentido de facilitar a rotina diária das pessoas, este trabalho tem como foco desenvolver um alimentador automatizado para animais domésticos, armazenando o alimento de forma adequada e fornecendo-o na dose correta e em intervalos determinados com a função de fornecer ao dono do animal a quantidade de alimento armazenado ainda disponível no dispositivo por meio de aplicativo em dispositivos móveis.

O dispositivo possibilita ao dono do animal até mesmo se ausentar de sua residência por alguns dias, a depender da capacidade de armazenamento, podendo incluir nesse pacote o monitoramento por vídeo do ambiente, pela mesma conexão de rede sem fio utilizada pelo alimentador.

Os dispositivos IoT para uso doméstico disponíveis atualmente no mercado possibilitam a tarefa automatizada de alimentar os animais domésticos com uma tomada com interruptor controlado por rede sem fio que estivessem conectados a algum assistente pessoal do sistema Android ou IOS, que teriam a função de acionar o interruptor por um tempo determinado, em intervalos de tempo determinados. Entretanto, o uso desse interruptor controlado por rede sem fio não possibilitaria verificar o nível de armazenamento de alimento, necessitando de outro dispositivo para tal função, que não está disponível no mercado.

Dessa forma, no intuito de integrar as duas funções, optou-se pela utilização da placa controladora Raspberry Pi, de baixo custo, com conectividade sem fio, com baixo consumo de energia e com funções básicas de programação, o que possibilita a integração das funções de fornecer o alimento na dose correta, no horário correto, fornecer a informação da quantidade de alimento armazenado e alertar o dono do animal quando o nível desse alimento armazenado estiver baixo ou nulo, além de possibilitar a implantação de outras funcionalidades.

2. DESENVOLVIMENTO

2.1 OBJETIVOS

O presente trabalho tem como objetivo geral projetar um distribuidor automático de alimentos para animais domésticos programável com base em Raspberry Pi e controlado via aplicativo móvel e tem como objetivos específicos:

- Realizar o desenho da estrutura física do protótipo de acordo com os parâmetros para criação de um produto.
- Projetar e implementar o circuito eletrônico que é acoplado a placa do Raspberry Pi e que permite o controle do sistema de acordo com os requisitos físicos do projeto.
- Desenvolver o código responsável pelo controle do hardware acoplado ao Raspberry Pi em linguagem de programação Python.
- Desenvolver um aplicativo móvel que permita a comunicação entre a placa do Raspberry Pi e um Smartphone.
- Verificar o funcionamento do protótipo em um ambiente real.

2.2. JUSTIFICATIVA E DELIMITAÇÃO DO PROBLEMA

O presente trabalho tem como problema de pesquisa “como criar um alimentador de animais domésticos automático, programável e com capacidade de monitoramento e operação remotos?”.

O crescimento populacional leva ao surgimento de necessidades que devem ser satisfeitas com o apoio da tecnologia, devendo ser proposto um desenvolvimento obrigatório em todos os seus campos de atuação para buscar soluções que tragam inovação e agilização. Vivemos uma era

em que o ser humano deseja realizar um grande número de tarefas com rapidez e eficiência, portanto as soluções devem ser práticas, simples, eficientes e oportunas.

Animais de estimação fazem parte desse crescimento. A estreita relação entre animais de estimação e ser humano está se tornando cada vez mais evidente, passando de animal de estimação a parte fundamental do núcleo familiar, por isso, os seus donos procuram uma forma de fornecer soluções tecnológicas às suas necessidades. Muitas vezes as pessoas donas de animais sentem uma enorme preocupação ao deixar seus lares e não ser capaz de levá-los, devido a essa situação, geralmente tentam realizar suas atividades no menor tempo possível, para que possam retornar e não sentir que estão abandonando seus animais de estimação.

Um fator decisivo é a alimentação, já que a comida deve ser constantemente fornecida, e em certas porções e deve ser servida em horários específicos, e em muitos casos podem ser esquecidas ou servidas de qualquer forma para se ganhar tempo ou não perder algum compromisso.

Além do problema de armazenamento e distribuição em que as pessoas colocam os alimentos em locais não adequados e em alguns casos no chão, podendo causar a proliferação de bactérias e contato com outros animais como ratos e pombos, com isso transmitem doenças aos cães e gatos, reduzindo a expectativa de vida do pet.

Para evitar todas essas preocupações e dar-lhes uma solução, propõe-se construir um dispensador de ração controlado por uma aplicação móvel, permitindo que as pessoas alimentem seus animais de estimação sem a necessidade de estar presente nas casas, apenas acesso a internet e alguns comandos usando um smartphone.

- Os animais são importantes para a vida dos seres humanos, como companhia e ajudantes como cão guia;
- Conectar esses dispositivos trará facilidade para os donos e melhora na vida dos animais.
- O armazenamento correto trará a diminuição da população de pequenos roedores como ratos e diminuição na proliferação de doenças como a leptospirose.

2. 3. FUNDAMENTAÇÃO TEÓRICA

A computação ubíqua tem como objetivo tornar a relação entre as pessoas e as máquinas tão integrada ao ponto em que se torne invisível, no sentido de “usar sem perceber”. O uso de tecnologias como o reconhecimento de voz e imagem e o desenvolvimento de sensores fotoelétricos, de temperatura, de distância, de pressão e de luminosidade, dentre muitos outros,

possibilitou que as máquinas passassem a reagir ao ambiente, incluindo-se nesse contexto a tomada de decisões com o uso de inteligência artificial. Nesse cenário, as coisas passaram a ser “inteligentes”, por reagirem ao ambiente e às pessoas.

> Num nível mais básico, ela ainda interliga computadores. Mas ela evoluiu para ser muito mais do que isso! Mark Weiser, um importante cientista da computação criou, no final da década de 1980, um conceito interessante. Nas palavras dele, “A computação ubíqua é a terceira onda da computação, que está apenas começando. Primeiro tivemos os mainframes, compartilhados por várias pessoas. Estamos na era da computação pessoal, com pessoas e máquinas estranhando umas às outras. A seguir vem a computação ubíqua, a era da tecnologia 'calma', quando a tecnologia recua para o pano de fundo de nossas vidas." É dele ainda a afirmação de que “As tecnologias mais importantes são aquelas que desaparecem. Elas se integram à vida do dia a dia, ao nosso cotidiano, até serem indistinguíveis dele." (NIC.BR, 2014).

O conceito de Internet das Coisas (*Internet of Things - IoT*) se caracteriza por uma grande quantidade de aparelhos conectados em rede, onde interagem entre si e com uma central controladora, que dá comandos e registra os eventos de forma a criar uma base de dados que pode ser utilizada para análise quantitativa e/ou qualitativa para uma finalidade.

Um bom conceito de IoT é apresentado no site da MapLink, empresa de serviços de geolocalização, roteirização, mapas e logística: “... IoT seria a tecnologia que transforma simples objetos em dispositivos capazes de se conectar com outras coisas, objetos e pessoas de forma a trocarem informações, através de conexões como internet, wifi, Bluetooth, etc.”

A primeira geração de dispositivos IoT voltados ao consumidor final foram os vestíveis, que ficavam em contato com o usuário, coletando dados e comunicando-se com outros dispositivos por meio de uma rede.

Aos poucos foram surgindo novos dispositivos que podiam ser controlados remotamente, realizando tarefas do cotidiano ao comando do usuário ou por meio de programação, iniciando as rotinas por um gatilho de tempo ou de algum outro evento, normalmente detectado por sensores, proporcionando uma experiência de automação na execução, seja no simples ato de realizar uma tarefa ou até de emitir alarmes de quaisquer problemas que possam vir a ocorrer.

2.3.1. Internet of Things (IoT) – Internet das coisas

A Internet das Coisas, traduzido do inglês *Internet of Things* (*IoT*), emergiu nas últimas décadas devido aos avanços de várias áreas, entre elas: sistemas embarcados, microeletrônica, comunicação, sensoriamento etc. Devido a isso, a *IoT* tem ganhado muita atenção tanto dos centros de pesquisas quanto dos diferentes setores industriais; decorrente da potencialidade de aplicação em diversas áreas da sociedade humana. Nesse sentido, a IoT pode ser definida de forma simplista

como uma extensão da internet atual, de modo a proporcionar aos objetos, equipamentos e dispositivos do nosso dia a dia, porém com maior capacidade computacional e de comunicação se conectarem à internet.

A conexão dos diferentes sistemas e equipamentos com a rede mundial de computadores tem por finalidade viabilizar, (i) controlá-los remotamente e (ii) permitir que os mesmos sejam acessados como provedores de serviços. Desse modo, essas novas habilidades e diretrizes geram muitas oportunidades em diferentes setores acadêmicos e industriais. No entanto, essas inúmeras possibilidades apresentam riscos e podem causar vários desafios técnicos e sociais.

Nesse sentido, a IoT tem modificado pouco a pouco os conceitos a respeito de redes de computadores, isso porque, pode ser observado a evolução dos conceitos ao longo do tempo. Por exemplo, segundo Kurose e Ross o termo redes de computadores começa soar um pouco envelhecido decorrente da ampla quantidade de equipamentos e tecnologias não tradicionais que são usadas na Internet (Kurose e Ross 2012). Já Tanenbaum et al. argumenta que a rede de computadores é um conjunto de computadores autônomos interconectados por uma única tecnologia (Tanenbaum 2002). Peterson e Davie caracteriza que a principal propriedade das redes de computadores têm sido a sua generalidade, ou seja, as mesmas são construídas sobre dispositivos de escopo geral e não são otimizadas para fins específicos tais como as redes de telefonia e TV (Peterson e Davie 2011).

Nos dias atuais, não só computadores comuns estão conectados à rede de computadores, assim como uma grande variedade de equipamentos, entre eles: laptops, smartTVs, smartphones, automóveis, consoles de jogos, webcams, entre outros. Usando os recursos desses diferentes dispositivos será possível detectar seu contexto, controlá-lo, bem como viabilizar troca de informações uns com os outros, acessar serviços da Internet e também interagir com pessoas. Essa nova perspectiva abre uma ampla gama de novas aplicações, como, por exemplo: em cidades inteligentes (*Smart Cities*), casas inteligentes (*Smart Home*), na área da saúde (*Healthcare*) etc.; por outro lado, os desafios também podem emergir como: as regulamentações, a segurança e as padronizações. Esse último é um dos pontos mais importantes e um dos elementos cruciais para o sucesso da IoT, principalmente em termos da padronização das tecnologias. Isso torna-se importante visando, pois permitirá que os diferentes dispositivos conectados à internet cresçam cada vez mais, de modo a tornar a IoT uma realidade. É nesse sentido que Mattern e Floerkemeier fazem uma analogia da IoT com a Web nos dias de hoje, no qual brevemente as coisas poderão ter habilidades de comunicação umas com as outras, podendo prover e usar serviços, de modo que fornecerão dados e poderão reagir a eventos.

Por fim, outra definição mais técnica, é que IoT tem sido vista como uma pilha de protocolos utilizados nos objetos inteligentes - isto porque esses objetos possuem capacidade de comunicação e processamento aliados a sensores, os quais transformam a utilidade destes objetos. Na IoT, eventualmente, a unidade básica de hardware pode apresentar no mínimo uma das seguintes características: (a) unidade(s) de processamento; (b) unidade(s) de memória; (c) unidade(s) de comunicação e; (d) unidade(s) de sensor(es) ou atuador(es) (Loureiro et al. 2003). Desse modo, os dispositivos com essas propriedades são denominados de objetos inteligentes (*Smart Objects*). Os objetos, ao estabelecerem comunicação com outros dispositivos, manifestam o conceito de estarem em rede, como discutido anteriormente.

2.3.2. Blocos Básicos da IoT

A IoT pode ser definida como a combinação de várias tecnologias, que são complementares de forma a viabilizar a integração dos objetos e equipamentos no ambiente físico ao mundo virtual. Os blocos básicos de construção da IoT são ilustrados na Figura 1, sendo eles: identificação, sensores/atuadores, comunicação, computação, serviços e semântica.

Figura 1 - Blocos básicos de construção da IOT

Fonte: SANTOS, B. P. S. et al (2017).

Na identificação, que é um dos blocos mais importantes, isso porque é primordial identificar os objetos unicamente para conectá-los à Internet. Sendo assim, as tecnologias tais como: RFID, NFC (Near Field Communication) e endereçamento IP podem ser utilizados visando identificar os objetos. Já os Sensores/Atuadores tem a finalidade de coletar as informações sobre o contexto em que os objetos se encontram. No bloco comunicação são utilizadas as diversas técnicas para conectar objetos inteligentes, ou seja, desempenhando um papel importante quanto ao consumo de energia dos objetos, desse modo o torna um fator crítico. Algumas das tecnologias usadas e que podem ser destacadas são Bluetooth, IEEE 802.15.4, WiFi e RFID. A computação inclui a unidade de processamento, tais como: o microcontrolador, os processadores e os FPGAs, com a finalidade de executar algoritmos locais nos objetos inteligentes (SANTOS, B. P. S. et al.).

No bloco de construção serviços - a IoT pode prover diversas classes de serviços, como, por exemplo: os serviços de identificação que são responsáveis por mapear as entidades físicas (EF) (de interesse do usuário) em entidades virtuais (EV) - por exemplo, a temperatura de um local físico em seu valor, coordenadas geográficas do sensor e instante da coleta. Além dos serviços de agregação de dados que visam coletar e sumarizar os dados homogêneos/heterogêneos que são obtidos dos objetos inteligentes. Os serviços de colaboração e inteligência visam agir sobre os serviços de agregação de dados, com a finalidade de tomar decisões e reagir de modo adequado a um determinado cenário. Nos serviços de ubiquidade tem por objetivo fornecer serviços de colaboração e inteligência em qualquer momento e qualquer lugar que sejam necessários (SANTOS, B. P. S. et al).

Por fim a semântica, esse bloco, se refere a habilidade de extração de conhecimento dos objetos na IoT, ou seja, trata da descoberta de conhecimento e uso eficiente dos recursos ou dados existentes, com o objetivo de prover determinado serviço. Desse modo, uma variedade de técnicas pode ser utilizada, entre elas: Resource Description Framework (RDF), Web Ontology Language (OWL) e Efficient XML Interchange (EXI) (SANTOS, B. P. S. et al).

2.4. METODOLOGIA

Com o estabelecimento do tema do presente Projeto Integrador 7, do ano de 2021, 15º Bimestre do Curso de Engenharia de Computação da Univesp, a equipe de trabalho reuniu-se para definir o projeto a ser desenvolvido.

O tema “Internet das Coisas” é bastante abrangente, e optou-se pela criação de um dispositivo alimentador de animais domésticos automatizado e programável, com possibilidade de monitoramento remoto e de ação remota, indo ao encontro do tema proposto.

Estabelecido o projeto a ser desenvolvido, e seguindo a metodologia do Design Thinking, foram entrevistados donos de animais domésticos que tinham como um problema a guarda de seus *pets*. De acordo com os entrevistados, sempre há a preocupação de com quem deixar a guarda, ou seja, levar o animal até a casa do cuidador, ou ceder a chave da área externa da residência para que o cuidador possa alimentar o animal, ou ainda, pagar a estadia dos animais em hotéis para pets (normalmente, clínicas veterinárias de nossa cidade que oferecem esse serviço adicional).

Além do custo financeiro, há o custo do risco de ceder a chave de seu imóvel, o risco de causar incômodo a terceiros, ou ainda, no pior dos casos, o risco de que o animal, dependendo de seu porte, acaba agredindo o cuidador, pela ausência do dono.

Foi mencionado pelos entrevistados de que a comodidade de que a alimentação de seus animais de estimação automatizada pode proporcionar, com a vantagem de que o alimento seria sempre disponibilizado aos animais em horários regulares, o que nem sempre é possível realizar devido às rotinas (ou a falta dela) do cotidiano das pessoas.

A associação de um dispositivo alimentador programável com uma câmera IP, equipamento disponibilizado no mercado por várias marcas, foi mencionada por 80% dos entrevistados, o que demonstra que é um projeto que atenderia tão bem às necessidades do público-alvo que já inspirava planos de como utilizar o equipamento no dia-a-dia.

As características mais desejadas pelo público-alvo foram:

- a capacidade de regular a quantidade oferecida por porção;
- a capacidade de programar os horários de fornecimento da alimentação (várias vezes por dia, se necessário);
- a capacidade de oferecer uma porção a qualquer momento, de forma remota;
- a capacidade de informar o nível de armazenamento de ração no reservatório;
- a capacidade de alertar sobre o baixo nível ou ausência de alimento no reservatório.

2.4.1. Prototipagem: componentes mecânicos

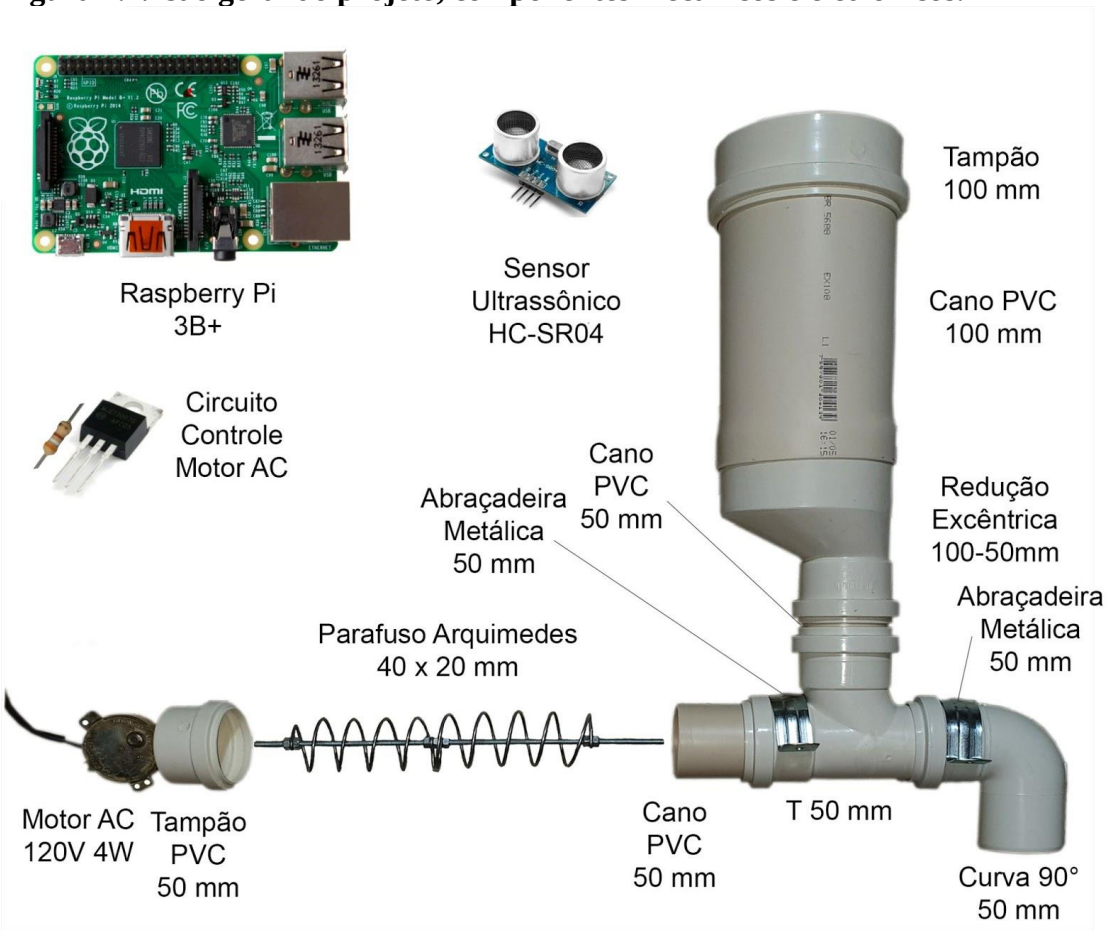
Considerando as características de comportamento dos animais comumente adotados como de estimação, optou-se por um projeto mecânico composto de um parafuso de Arquimedes, instalado no interior de um tubo de PVC de forma que, ao ser acionado, realiza o transporte de ração do reservatório, localizado acima da tubulação, para a extremidade oposta, cuja terminação

se dá um ângulo reto para possibilitar a ancoragem do parafuso de Arquimedes e impossibilitar que os animais, ao inserirem suas patas ou a língua em busca de obter quantidade extra de alimento, não acabem enroscando seus membros no parafuso de Arquimedes. Optou-se também pelo diâmetro de 50 mm, pequeno o bastante para evitar esse tipo de ocorrência.

> "O chamado parafuso de Arquimedes é um mecanismo bastante antigo, que vem sendo utilizado desde as mais remotas civilizações como dispositivo para transportar diversos tipos de materiais de um nível para outro. Trata-se simplesmente de uma rosca embutida em um tubo. Mergulhando-se uma de suas extremidades no material a ser transportado, e girando-se o conjunto, o material é obrigado a entrar pela rosca e subir ao longo do eixo, até transbordar na parte superior." (MATOS, F. C. et al, 2013)

Conforme se observa na Figura 2, o parafuso de Arquimedes foi construído com arame galvanizado com 2 mm de espessura, em formato helicoidal de 4 cm de diâmetro e passo de 18 mm, acoplado em uma barra roscada de 4 mm, que funciona como estrutura do parafuso de Arquimedes. O arame helicoidal foi confeccionado em 2 segmentos de 4,5 voltas cada, com a finalidade de oferecer uma estrutura menos flexível e com resistência suficiente às forças de peso e de atrito da ração com as paredes do tubo de transporte.

Figura 2. Visão geral do projeto, componentes mecânicos e eletrônicos.



Fonte: Os autores, 2021.

Na parte inicial do canal de transporte, também fechado, por sua vez, por um tampão, é ancorada a outra extremidade do eixo do parafuso de Arquimedes, no qual é conectado o motor elétrico AC, de baixa velocidade, sendo outro recurso com a finalidade de prevenir acidentes com os animais.

O tubo de transporte é composto basicamente de conexões de tubos de PVC 50 mm, com insertos de tubo PVC 50 mm apenas com a função de ligação entre essas conexões, constituindo-se de um tampão, um inserto de cano PVC, um T, e uma curva de 90°.

Ainda na Figura 2, observa-se que o reservatório de ração se compõe de uma redução excêntrica de 100 mm para 50 mm, acoplada ao T 50 mm por meio do inserto de cano PVC 50 mm, um pedaço de 22 cm de cano 100 mm acoplado na extremidade mais larga; um tampão de 100 mm faz o papel de tampa do reservatório, onde será instalado o sensor de nível de ração.

Todo o mecanismo é ancorado em uma base de compensado laminado naval, de 20 mm de espessura, por meio de abraçadeiras metálicas para fixação de canos de 50 mm de diâmetro.

2.4.2. Prototipagem: componentes elétricos e eletrônicos

Para atender as características apontadas pelos entrevistados, optou-se por utilizar uma placa controladora programável de baixo custo, o Raspberry Pi, bastante conhecido no mercado de equipamentos para aprendizagem de disciplinas como Circuitos Elétricos, Programação de Computadores e Eletrônica, dentre outras.

O uso do Raspberry Pi combinado com um sensor de ultrassom para medição de distância pode fornecer dados sobre o nível de ração armazenada no dispositivo, enquanto o uso de um micro-motor controlado por tempo de operação pode fornecer a dosagem correta. A programação dos horários de alimentação pode ser feita na interface do controlador, que pode, também informar via rede sem fio o nível de armazenamento de ração, o horário em que as refeições foram oferecidas e enviar alertas sobre um baixo nível de armazenamento de ração, atendendo às principais características desejadas.

Adicionalmente, uma placa controladora pode controlar mais de um alimentador, dada a quantidade de entradas e saídas do dispositivo.

2.4.3. Raspberry Pi

Decidimos utilizar como plataforma de controle de operação para o alimentador de animais um Raspberry PI 3 B+, que, como pode ser visto na Figura 3, é uma placa de tamanho e de consumo

de energia reduzidos, medindo aproximadamente o tamanho de um cartão de crédito e consumindo aproximadamente 2,5 kW/h por mês, e que pode ser usado como um computador de placa única. Isto permite ter um baixo custo e, adicionalmente, uma excelente capacidade de processamento de dados. Esta placa foi criada pela Raspberry Pi Foundation em 2012, com o objetivo de simplesmente ensinar tópicos relacionados à programação em escolas do Reino Unido.

Figura 3: Raspberry Pi 3 B+.

Fonte: Raspberry Pi Foundation, 2018.

O Raspberry Pi é uma placa que, como mencionado acima, é usada para ensinar noções básicas de programação, e seu baixo custo permite que pessoas com pouco recursos acessem a plataforma para praticar linguagens de programação nativas, como o Python.

O software utilizado nesta placa é baseado no kernel Linux denomidado Raspbian, que é uma variante da distribuição Debian baseada no ARM hard-float, sendo um porte da arquitetura Wheezy otimizada para o conjunto de instruções ARMv6 do hardware do Raspberry Pi.

O Raspbian tem uma experiência de uso bastante intuitiva e fácil de configurar, semelhante ao Linux Ubuntu. Para programar pelo Raspbian não é necessário a utilização de monitor, podendo ser acessado através de SSH (Secure Socket Shell), que é um protocolo de rede que permite aos usuários acessar e gerenciar servidores pela internet.

No presente trabalho iremos utilizar o modelo Raspberry Pi 3 B+, que, conforme explicado pela Raspberry Pi Foundation (tradução dos autores) possui as seguintes especificações:

- Processador Quad Core 1.2 GHz Broadcom BCM2837 64-bit
- 1 Gb de memória RAM
- Conexão Wireless BCM43438
- Bluetooth Low Energy (BLE) on-board

- Ethernet: 10/100 mbps
- 40 Pinos GPIO (General Purpose Input/Output)
- 4 portas USB 2.0
- Saídas RCA: estéreo de 4 polos e porta de vídeo composto
- Porta HDMI
- Porta da câmera: CSI para conectar uma câmera Raspberry Pi
- Porta de exibição: DSI para conectar uma tela sensível ao toque do Raspberry Pi
- Porta Micro SD: para carregar o sistema operacional e armazenar dados

Dentre as especificações acima elencadas, a conexão de rede sem fio presente na placa é a que permitirá ao Raspberry Pi a capacidade de estar conectado à Internet e possibilitar a notificação ao dono do animal informações sobre o nível de ração armazenada, os horários e a quantidade em que a alimentação foi servida.

Com o uso do Raspberry Pi 3 B+, é possível controlar:

- - um micro-motor de rotação fixa, como ilustrado na Figura 4, para movimentar o mecanismo de transporte de ração. O horário de acionamento pode ser controlado por uma agenda, que controlará o horário de ligar e de desligar o motor, várias vezes ao dia. A quantidade servida pode ser controlada pelo tempo em que o motor permanecer acionado, uma vez que a rotação do motor é fixa e constante.
- - um sensor ultrassônico HC-SR04, como pode ser visto na Figura 5, que tem a capacidade de medir a distância até um obstáculo ou objeto. Instalado na parte superior do reservatório de ração, voltado para baixo, o sensor pode medir a distância até a ração, o que resultaria no nível de ração consumida. Sendo conhecida a altura total do reservatório, a diferença resultará no nível de ração restante no reservatório. Por programação, o Raspberry Pi 3 poderá enviar uma notificação ao smartphone do dono do animal, alertando sobre o baixo nível de ração no reservatório.

Figura 4. Motor de prato giratório de fornos de microondas. Visão frontal e traseira.

Fonte: Os autores, 2021.

Figura 5. Sensor ultrassônico HC-SR04.

Fonte: Os autores, 2021.

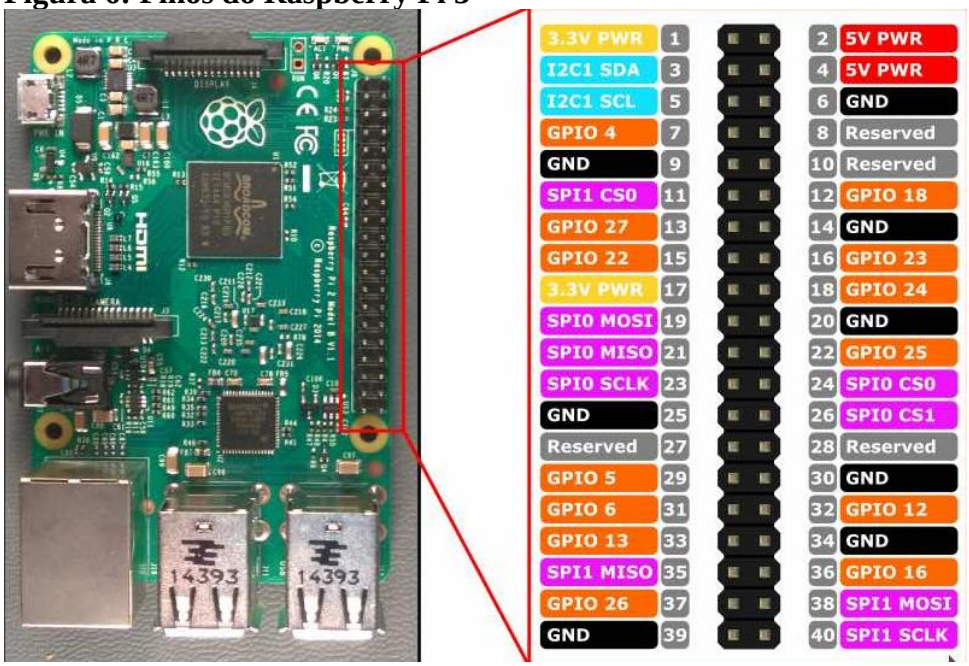
A programação do Raspberry Pi nos permite atender à necessidade dos usuários:

- na capacidade de regular a quantidade oferecida por porção, por meio do controle do tempo de funcionamento do motor;
- na capacidade de programar os horários de fornecimento da alimentação, várias vezes por dia, se necessário, pelo agendamento dos períodos de acionamento, via programação;
- na capacidade de oferecer uma porção a qualquer momento, de forma remota, por meio de aplicativo executado em smartphones;
- na capacidade de informar o nível de armazenamento de ração no reservatório, por meio dos dados fornecidos pelo sensor de ultrassom;
- na capacidade de alertar sobre o baixo nível ou ausência de alimento no reservatório, por meio de programação do Raspberry Pi, tendo como gatilho um determinado nível de ração detectado pelo sensor de ultrassom.

2.4.4. Os pinos GPIO da Raspberry Pi

A placa Raspberry Pi 3, como ilustrado na Figura 6, possui 40 pinos de entrada e saída. Dentre elas, existem pinos de entrada e saída de uso geral, configuráveis, chamadas de GPIO (*General Pupose In-Out*), que permitem a conexão entre o Raspberry Pi e os diferentes módulos e dispositivos com os quais se pode trocar dados. As demais portas (pinos) são de uso específico, como interfaces UART, sendo utilizados dois pinos (RX e TX), com os quais podem ser implementadas comunicações seriais, que podem ser muito úteis para se conectar com outras placas, como o Arduino. Outros pinos especiais também são usados para implementar protocolos de comunicação I2C (pinos SCL e SDA). O Raspberry Pi também possui protocolo de comunicação SPI, aos quais dedica pinos MOSI, MISO SCLK CE0 e CE1.

Figura 6: Pinos do Raspberry Pi 3



Fonte: SUHANKO, 2021.

2.4.5. Funcionamento do Circuito

Aos pinos GPIO podem ser atribuídos estados lógicos de HIGH e LOW, cujos valores de voltagem são de 3,3 volts e 0,0 volts, respectivamente. O limite de corrente elétrica dos pinos GPIO é da ordem de 50 miliamperes, que, aliada a baixa tensão, não fornece potência suficiente para que o motor funcione, podendo, inclusive, danificar a placa caso se realize a conexão.

Além da questão da baixa potência suportada pela GPIO, o motor utilizado no presente projeto é um motor síncrono de corrente alternada, 120 V, 60 Hz , 4 W, rotação de 6 rpm (Figura 7). Esse motor é comumente utilizado em pratos giratórios de fornos microondas e grades direcionadoras de fluxo de circuladores de ar domésticos. É um motor de baixo custo, com torque suficiente para movimentar o parafuso de Arquimedes do sistema de transporte de ração, além de girar em baixa rotação, o que previne que ocorram acidentes com os animais.

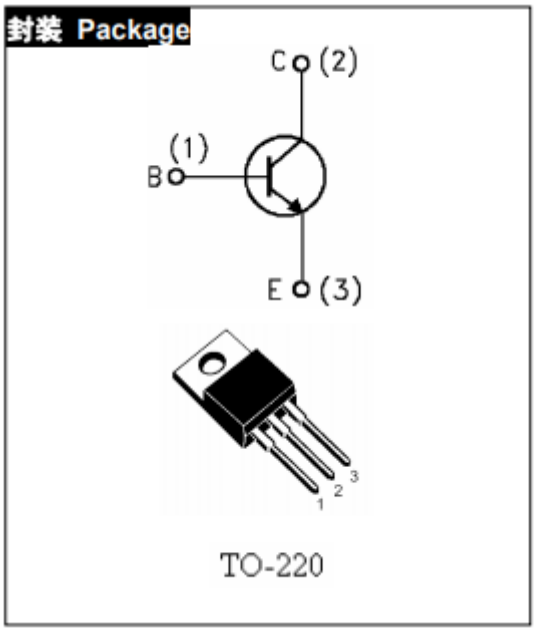
Figura 7: Motor AC, 120 V, 4 W, 6 rpm.

Fonte: Os autores, 2021.

Para solucionar o problema, foi confeccionada uma placa eletrônica que acionará o fornecimento de corrente alternada para o motor.

O projeto de controle do Motor é baseado em um transistor D13007K, conforme o ilustrado na Figura 8, em modo interruptor; dessa forma, quando a placa Raspberry Pi envia um sinal de HIGH através de um de seus pinos GPIO para o pino Base do transistor, o mesmo acionará a passagem de corrente entre o coletor ligado a uma fonte de corrente de 127 V e o emissor, onde encontra-se o motor e o pino ligado ao Terra, fazendo com que a corrente passe pelo circuito livremente, girando o motor e acionando o tubo de transporte de ração para o recipiente do alimentador; quando a placa Raspberry Pi envia um sinal de LOW para o pino Base do transistor, ocorre o inverso, com o corte da passagem de corrente e desligamento do motor.

Figura 8: Transistor D1300K7



Fonte: Jilin Sino-Microeletronics, 2009.

Para calcular a resistência de base do transistor, a seguinte fórmula foi usada:

$$\text{Rbase} = \frac{(\text{volt} - 1{,}5)}{\frac{Cte}{HFE}}$$

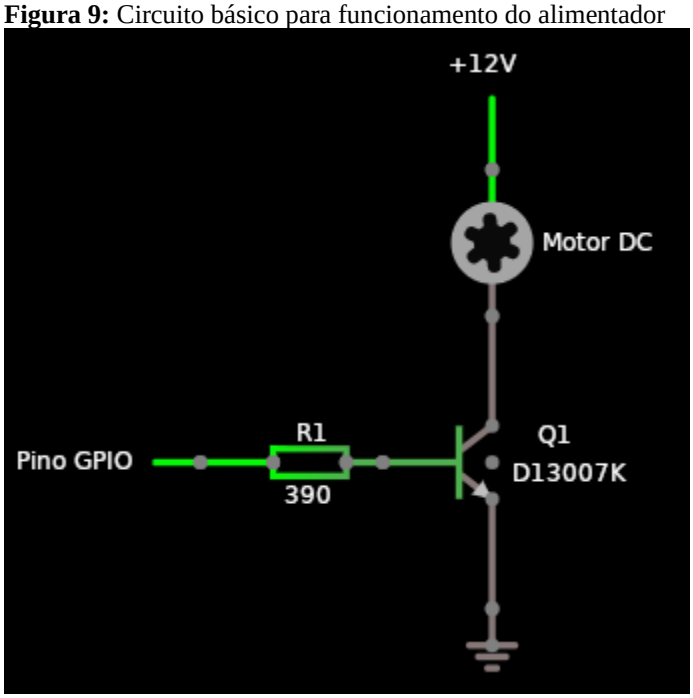
Onde:

- *Volt* é a voltagem aplicada na base do transistor, neste caso corresponde aos 3,3 volt gerados pelo pino do Raspberry Pi.

- O valor *1,5* corresponde à tensão base-emissor, queda de tensão, típica neste tipo de transistores; este é um valor obtido no datasheet do fabricante.
- *Cte* é a corrente que passará pelo transistor de seu coletor ao emissor.
- *HFE,* corresponde ao valor de ganho de corrente, comumente conhecido com Beta do transistor.

O valor calculado neste caso não corresponde ao valor de uma resistência comercial, por isso será utilizado o valor de resistência mais próximo.

Figura 9: Circuito básico para funcionamento do alimentador



Fonte: Os autores, 2021.

2.4.6. Interação entre componentes mecânicos e eletrônicos

O sensor ultrassônico HC-SR04, instalado na tampa do reservatório de ração mede a distância do sensor ao nível de ração. Sendo conhecido o tamanho do reservatório de ração, que consiste de um redutor excêntrico em PVC de 100 mm para 50 mm e um cano em PVC de 100 mm, o sensor fornece a informação do nível de ração remanescente. Quando a distância aferida se torna maior que um determinado nível, o Raspberry Pi 3 comunica-se com o smartphone do dono do animal, avisando sobre o baixo nível de ração restante.

Nos horários programados, o Raspberry Pi 3 aciona o motor do sistema de transporte de ração por tempo determinado, fornecendo ao animal a quantidade correta, e envia ao smartphone do usuário a informação de que o evento ocorreu.

No aplicativo instalado no smartphone do usuário, é possível enviar uma ordem de execução do comando para fornecer uma porção extra de ração ao animal.

O projeto inicial previa um vibrador para o reservatório de ração composto por um motor de corrente contínua de 3 V de alta rotação (3600 rpm) ao qual seria acoplado um volante inercial excêntrico de 15 gramas, para produzir vibração no reservatório com a finalidade de facilitar o movimento da ração, mas que mostrou-se desnecessário já nos primeiros testes. A ideia do vibrador foi descartada em nome da contenção de custos do projeto.

O controle de acionamento do motor AC 120 V por meio da porta GPIO do Raspberry Pi logrou-se com êxito, acionando o motor adequadamente.

A construção do parafuso de Arquimedes com arame galvanizado de 2 mm e barra roscada 4 mm mostrou-se acertada, pelo baixo custo do material e pela resistência obtida com o uso de dois segmentos de 4,5 elos, que resistiu à tração da ração ao longo do sistema transportador sem deformações, inclusive em dias de clima úmido, em que o escoamento da ração tende a ser dificultado.

O uso do redutor excêntrico para compor o fundo do reservatório de ração, conforme se observa na Figura 10, mostrou-se acertado, pois seu formato facilita o escoamento do alimento ao tubo transportador e dispensa o uso do vibrador.

Figura 10: Redutor excêntrico 100-50 mm em PVC.

Fonte: Os autores, 2021.

2.4.7. Programação do Raspberry Pi 3 B+

O Raspberry Pi utiliza o sistema operacional Rasberry Pi OS, que é instalado em um cartão SD. Nessa interface, há o recurso de programação em linguagem Python 3, que é o software que iremos utilizar para controlar o dispositivo do presente projeto.

A placa foi programada com um código, que se pode ver na Figura 11, que atende às seguintes especificações:

1- Determinação da quantidade de ração por porção: a quantidade de ração é regulada pelo tempo em que o motor permanece acionado para cada porção.

2- Determinação dos horários de servir as porções ao longo do dia: é possível programar os horários em que as refeições dos animais devem ser servidas.

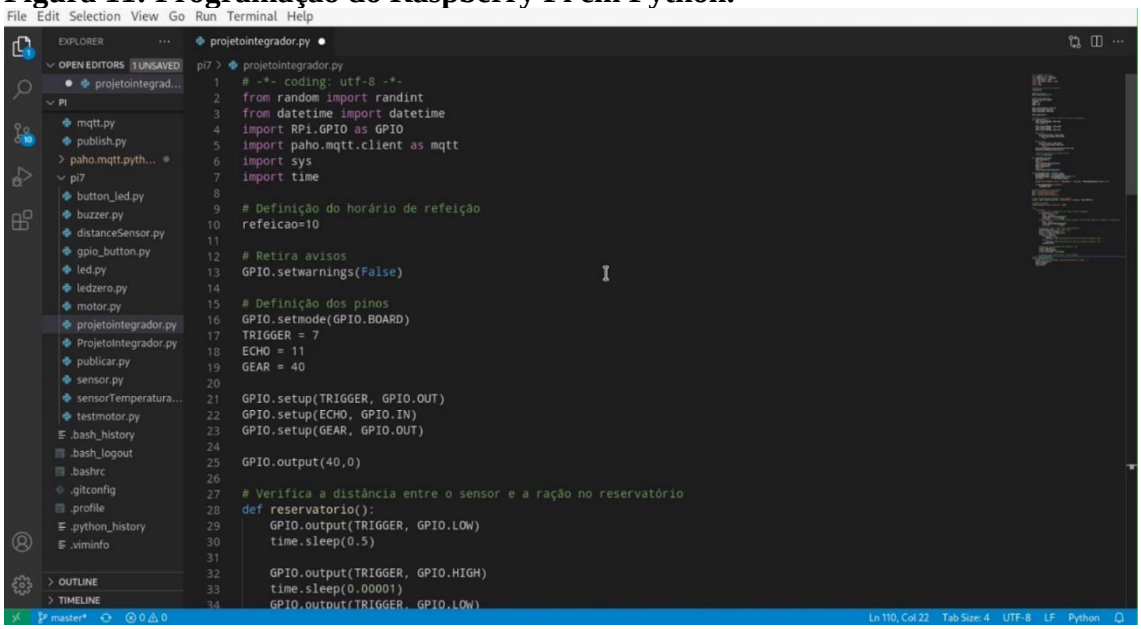
3- Servir uma porção extra a qualquer momento: é possível enviar um comando para que seja servida uma porção adicional, além dos eventos programados.

4- Monitoramento do nível de ração: é possível verificar o nível de ração do reservatório.

5- Definição do nível baixo de ração: é possível ajustar o nível de ração em que o Raspberry Pi irá alertar sobre o nível baixo de ração.

6- Envio de notificação de alerta de nível baixo de ração, por meio do protocolo MQTT, ao smartphone do dono do animal.

Figura 11: Programação do Raspberry Pi em Python.



Fonte: Os autores, 2021.

2.4.8. O Protocolo MQTT

Para a comunicação do dispositivo com os smartphones, foi utilizado o protocolo MQTT, desenvolvido pela IBM, que é um protocolo simples, leve e seguro de ser aplicado.

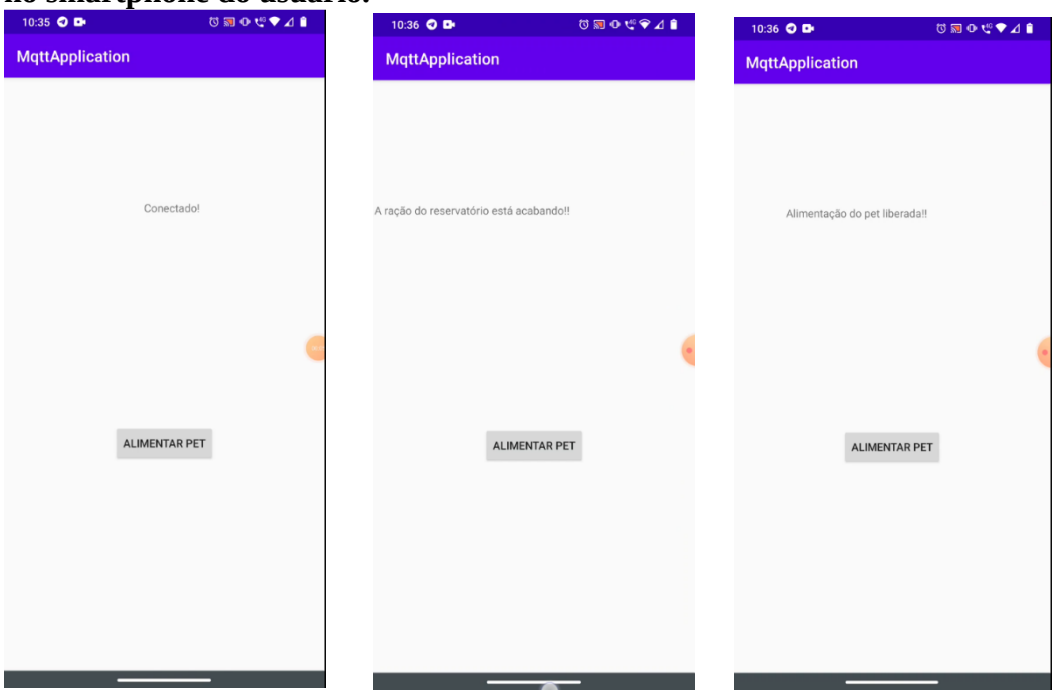
O MQTT (Messagem Queue Telemetry Transport - Transporte de Filas de Mensagem de Telemetria) foi desenvolvido com base em TCP/IP com o objetivo de comunicação entre máquinas com do tipo Publicação/Assinatura.

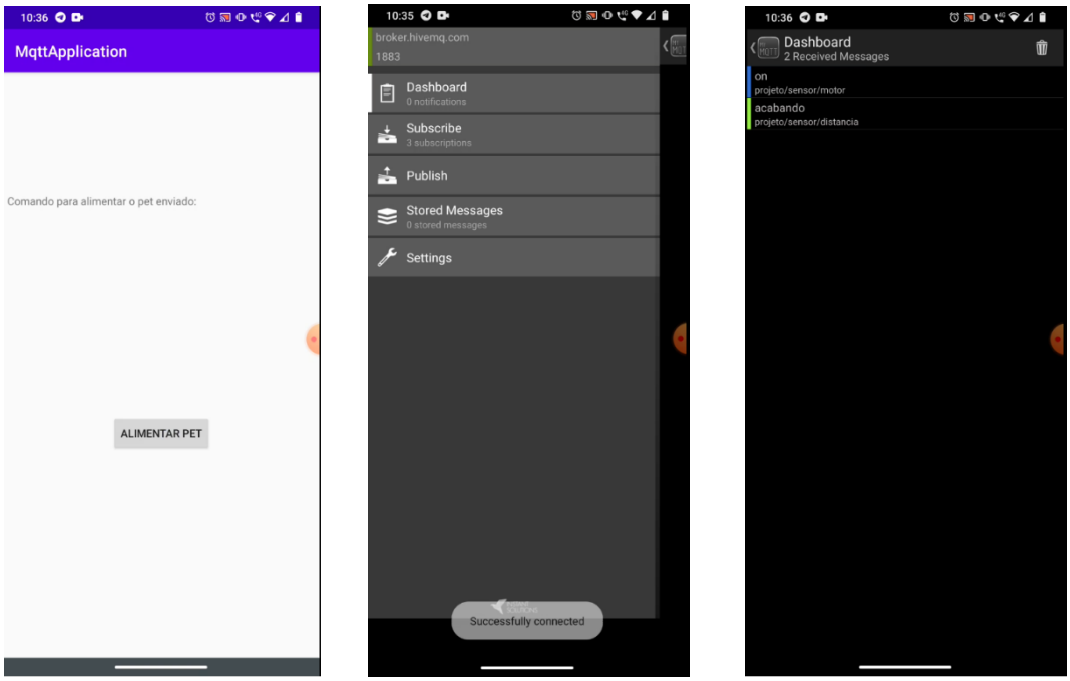
Criado em 1999 para aplicação proprietária, foi disponibilizado de forma gratuita ao público em 2010. Como vantagens de sua utilização, temos melhor qualidade de serviço, compatibilidade com diversas linguagens de programação, disponibilidade de serviço baseada em prioridades de mensagem e garantia de entrega, dentre outros.

> "O Protocolo de Comunicação MQTT se mostra a melhor opção para uso da Internet em projetos IoT, suas qualidades são inúmeras, valendo mencionar a possibilidade de monitoramento de diversos sensores de forma simultânea, até o seu suporte de comunicação assíncrona e baixo deslocamento de banda." (Locatelli, 2020)

Com o uso desse protocolo, foi realizada a conexão entre o smartphone do usuário e o Raspberry Pi, como se pode observar na Figura 12.

Figura 12: Telas de configuração do aplicativo (em azul) e do cliente MQTT (tons de cinza) no smartphone do usuário.

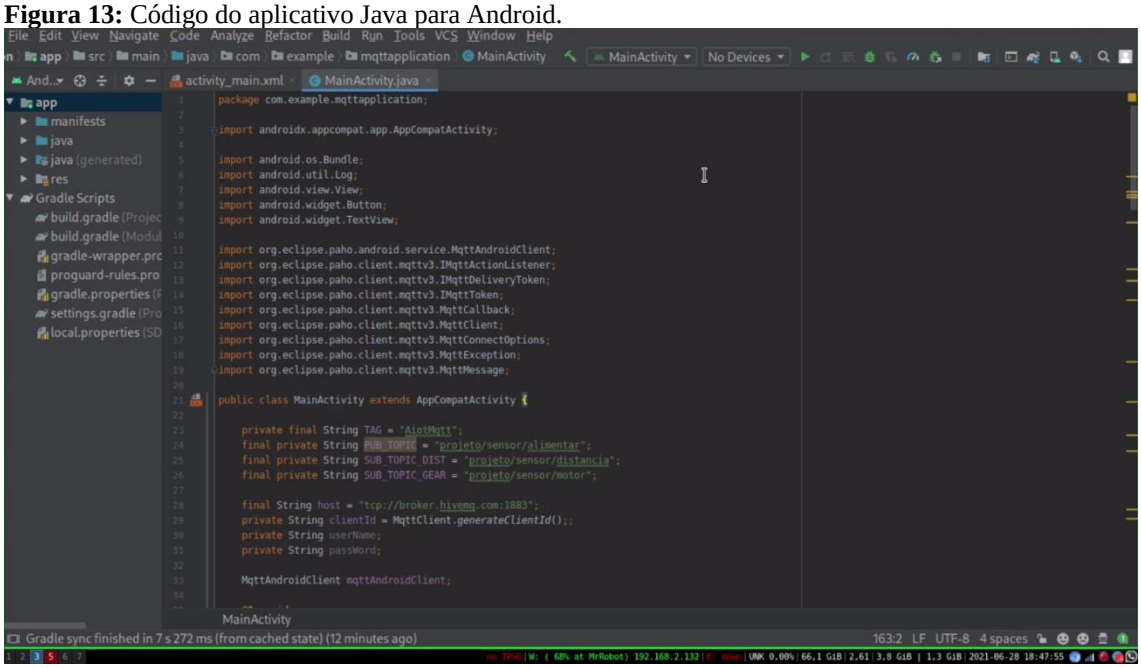




Fonte: Os autores, 2021.

O aplicativo elaborado para execução no Android foi escrito na linguagem Java, conforme se pode observar na Figura 13.

Figura 13: Código do aplicativo Java para Android.



Fonte: Os autores, 2021.

3. CONSIDERAÇÕES FINAIS

O uso da Internet possibilitou que, além de pessoas, as máquinas se comuniquem por esse meio, o que caracteriza que a máquina atende ao conceito de Internet das Coisas (*Internet of Things - IoT*).

O uso da metodologia de *Design Thinking* possibilitou atender a uma demanda dos donos de animais de estimação que, por diversos motivos, não podem estar presentes em suas residências para cumprir tarefas do cotidiano, como alimentar seus *pets*.

O presente trabalho buscou atender as necessidades levantadas na fase de projeto e planejamento junto à comunidade, que eram centradas em segurança, automação, interação e alerta ao usuário no que se refere à rotina de alimentação dos animais.

Com os testes realizados do protótipo junto à comunidade de donos de animais de estimação que nos forneceu as informações sobre suas demandas, pudemos constatar que o uso do Raspberry Pi permitiu atender aos últimos três quesitos, e o uso de motor de baixa velocidade e o design dos componentes mecânicos atendeu ao requisito de segurança dos animais.

Ficou claro também que o presente trabalho é passível de refinamentos e de ajustes para cortes de custo visando uma produção em série, que, a princípio, não seria seu objetivo.

Entretanto, diversos papéis alheios às necessidades inicialmente elencadas foram cumpridos, em especial o papel da placa controladora Raspberry Pi, criada com a finalidade de servir como instrumento de ensino de eletrônica e programação.

A interação com componentes mecânicos e a comunicação remota via aplicativo executado no smartphone do usuário foi enriquecedora, no sentido didático, aos membros da equipe do presente trabalho.

REFERÊNCIAS

A. Y. R. Hernández; R. C. Sánchez. **Conversión A/D con Protocolo SPI para Audiofrequências.** Departamento de Ingeniería Electrónica, División de Ingenierías Campus Irapuato - Salamanca, Universidad de Guanajuato. 2015.

ABNT – Associação Brasileira de Normas Técnicas. **NBR 10520**: Informação e documentação - Citações - Apresentação. Rio de Janeiro: ABNT, 2002.

ABNT – Associação Brasileira de Normas Técnicas. **NBR 14724**: Informação e documentação. Trabalhos Acadêmicos - Apresentação. 3. ed. Rio de Janeiro: ABNT, 2011.

ABNT – Associação Brasileira de Normas Técnicas. **NBR 6023**: Informação e documentação - Referências - Elaboração. 2. ed. Rio de Janeiro: ABNT, 2018.

ABNT – Associação Brasileira de Normas Técnicas. **NBR 6024**: Informação e documentação - Numeração progressiva das seções de um documento - Apresentação. 2. ed. Rio de Janeiro: ABNT, 2002.

BRANDÃO, Bruna. **O que é IoT - Como melhorar rotinas empresariais, industriais e pessoais com a internet das coisas?** MapLink, 2020. Disponível em:https://maplink.global/blog/o-que-e-iot/. Acessado em: 16 jun 2020.

E. UPTON; G. HALFACREE. **Raspberry Pi:** guia do usuário. 4. ed. Alta Books, 2017.

G. PATRIK; B. ANTÓNIA. **Opportunities of Raspberry Pi's Use in Education Raspberry Pi** felhasználási lehetőségei az oktatásban. Dennis Gábor College/Institute of Basic and Technical Sciences, Budapest, Hungary, 2018.

JILIN SINO-MICROELECTRONICS CO., LTD. **3DD13007K Datasheet**. 10 out 2009. Disponível em; https://www.datasheetq.com/datasheet-download/235348/1/Hwdz/D13007K. Acessado em: 10 mai 2021.

Kurose, J. F.; Ross, K. W. Computer Networking: A TopDown Approach (6th Edition). Pearson, 6th edition, 2012.

LOCATELLI, Carolina. **Introdução ao MQTT**. Curto Circuito. 14 jan. 2020. Disponível em: https://www.curtocircuito.com.br/blog/Categoria%20IoT/introducao-ao-mqtt. Acessado em 20 jun 2021.

Loureiro, A. A. et al. Redes de Sensores Sem Fio. In Simpósio Brasileiro de Redes de Computadores (SBRC), p. 179–226, 2003.

Mattern, F.; Floerkemeier, C. . From the internet of computers to the internet of things. In From active data management to event-based systems and more, Springer, p. 242–259, 2010.

MATOS, F.C. et al. O parafuso de Arquimedes: uma inovação no ensino de matemática sob a perspectiva de modelagem matemática no IFPA. In: ENEM - Encontro Nacional de Educação Matemática. 11.18 a 21 jul 2013, Curitiba. **Anais**... Curitiba: SBEM - Sociedade Brasileira de Educação Matemática, 2013. Disponível em: http://sbem.iuri0094.hospedagemdesites.ws/anais/XIENEM/pdf/2861_1186_ID.pdf. Acessado em: 11 jun 2021.

NIC.BR. **A Internet das coisas, explicada pelo NIC.br**. Publicado em 16/07/2014. Disponível em: https://www.nic.br/videos/ver/a-internet-das-coisas-explicada-pelo-nic-br/. Acessado em: 23/04/2021.

Peterson, L. L. Davie, B. S. Computer Networks, Fifth Edition: A Systems Approach. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 5th ed., 2011.

Raspberry Pi Foundation. **Raspberry Pi 3 Model B+**. 2018. Disponível em: https://www.raspberrypi.org/products/raspberry-pi-3-model-b-plus/. Acessado em: 03 jun 2021.

SANTOS, B. P. S. et al. Internet das coisas: da teoria à prática, Belo Horizonte. 4-15, 2017. Disponível em: https://homepages.dcc.ufmg.br/~mmvieira/cc/papers/internet-das-coisas.pdf. Acessado em: 05 jun. 2021.

SUHANKO, James. **Vários modos de interagir com o GPIO do Raspberry Pi**. Do Bit ao Byte. 29 nov. 2017. Disponível em: https://i1.wp.com/www.dobitaobyte.com.br/wp-content/uploads/2017/11/bmc_header_pinout.png. Acessado em: 9 mai. 2021.

APÊNDICES

Apêndice A – Programação do Raspberry Pi - Código em Python

```
# -*- coding: utf-8 -*-
from random import randint
from datetime import datetime
import RPi.GPIO as GPIO
import paho.mqtt.client as mqtt
import sys
import time

# Definição do horário de refeição
refeicao=10

# Retira avisos
GPIO.setwarnings(False)

# Definição dos pinos
GPIO.setmode(GPIO.BOARD)
TRIGGER = 7
ECHO = 11
GEAR = 40

GPIO.setup(TRIGGER, GPIO.OUT)
GPIO.setup(ECHO, GPIO.IN)
GPIO.setup(GEAR, GPIO.OUT)

GPIO.output(40,0)

# Verifica a distância entre o sensor e a ração no reservatório
def reservatorio():
GPIO.output(TRIGGER, GPIO.LOW)
        time.sleep(0.5)

GPIO.output(TRIGGER, GPIO.HIGH)
    time.sleep(0.00001)
GPIO.output(TRIGGER, GPIO.LOW)

            while True:
                        pulse_start_time = time.time()
                        if GPIO.input(ECHO)==GPIO.HIGH:
                                break

            while True:
                        pulse_end_time = time.time()
                        if GPIO.input(ECHO)==GPIO.LOW:
                                break
pulse_duration=pulse_end_time-pulse_start_time
distance=(34300*pulse_duration)/2
```

```
# Retorna a distância em formato inteiro
      return int(distance)

def ligarMotor(seconds):
      GPIO.output(40,1)
                  msg="on"
client.publish(psm,msg,qos=0)
     time.sleep(seconds)
      GPIO.output(40,0)
                msg="off"
client.publish(psm,msg,qos=0)

def on_message(client, userdata, msg):
MensagemRecebida = str(msg.payload)
MensagemRecebida = MensagemRecebida.strip("\'b")
print(psa + "/" + str(MensagemRecebida))

print("[MSG RECEBIDA] Topico: "+msg.topic+" / Mensagem:
"+MensagemRecebida.strip("\'b"))

if MensagemRecebida=="alimentar":
                                    ligarMotor(60)

# topicos providos por este sensor
psd = "projeto/sensor/distancia"
psm = "projeto/sensor/motor"
psa = "projeto/sensor/alimentar"

# cria um identificador baseado no id do sensor
client = mqtt.Client(client_id = 'NODE:5000-10', protocol = mqtt.MQTTv31)

# conecta no broker
client.connect("broker.hivemq.com", 1883)

try:
            while True:
                                    # verifica se o reservatorio está cheio e envia a mensagem
                                    if reservatorio() > 50:
                                              msg = "acabando"
                                              client.publish(psd,msg,qos=0)
                                              print(psd + "/" + str(msg))
                                    else: # Necessário para que não fique avisando o usuário mesmo
depois de completar o reservatorio
                                              msg = "cheio"
                                              client.publish(psd,msg,qos=0)
                                              print(psd + "/" + str(msg))

                                    # Verificar a hora e liga o motor caso necessite
                                    now=datetime.now() # data e hora atual
```

```
                        hora = int(now.strftime('%H'))
                        minuto = int(now.strftime('%M'))
                        if hora == refeicao:
                                if refeicao == 10:
                                        refeicao == 18 # definição de um novo horário para
alimentar o pet
                                else:
                                        refeicao == 10 # definição de um novo horário para
alimentar o pet
                                ligarMotor(60)

                        # Verifica se há mensagem para alimentar o pet
                        client.loop_start()
                        client.subscribe(psa)
                        client.on_message = on_message

                        # Tempo de espera para iniciar o loop novamente
                        time.sleep(3)
except KeyboardInterrupt:
print("\nCtrl+C pressionado, encerrando aplicacao e saindo...")
        client.disconnect()
           GPIO.cleanup()
                sys.exit(0)
```

Apêndice B – Aplicativo Android - Código em Java

```
package com.example.mqttapplication;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;

import org.eclipse.paho.android.service.MqttAndroidClient;
import org.eclipse.paho.client.mqttv3.IMqttActionListener;
import org.eclipse.paho.client.mqttv3.IMqttDeliveryToken;
import org.eclipse.paho.client.mqttv3.IMqttToken;
import org.eclipse.paho.client.mqttv3.MqttCallback;
import org.eclipse.paho.client.mqttv3.MqttClient;
import org.eclipse.paho.client.mqttv3.MqttConnectOptions;
import org.eclipse.paho.client.mqttv3.MqttException;
import org.eclipse.paho.client.mqttv3.MqttMessage;

public class MainActivity extends AppCompatActivity {

    private final String TAG = "AiotMqtt";
    final private String PUB_TOPIC = "projeto/sensor/alimentar";
    final private String SUB_TOPIC_DIST = "projeto/sensor/distancia";
    final private String SUB_TOPIC_GEAR = "projeto/sensor/motor";

    final String host = "tcp://broker.hivemq.com:1883";
    private String clientId = MqttClient.generateClientId();;
    private String userName;
    private String passWord;

    MqttAndroidClient mqttAndroidClient;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        final TextView tvStatus = (TextView)findViewById(R.id.tvStatus);
        //setContentView(tvStatus);

        /* MqttConnectOptions */
        MqttConnectOptions mqttConnectOptions = new MqttConnectOptions();
        //mqttConnectOptions.setUserName(userName);
        //mqttConnectOptions.setPassword(passWord.toCharArray());

        /* MqqtAndroidClient */
        mqttAndroidClient = new MqttAndroidClient(getApplicationContext(),host,clientId);
```

```
mqttAndroidClient.setCallback(new MqttCallback() {
    @Override
    public void connectionLost(Throwable cause) {
        Log.i(TAG, "connection lost");
    }

    @Override
    public void messageArrived(String topic, MqttMessage message) throws Exception {
        Log.i(TAG,"topic: " + topic + ", msg: " + new String(message.getPayload()));
        String msg = new String(message.getPayload());

        if(msg.toString().equals("acabando")){
            String aviso = "A ração do reservatório está acabando!!";
            tvStatus.setText(aviso);
            Log.i(TAG,aviso);
        }

        if(msg.toString().equals("on")){
            String aviso = "Alimentação do pet liberada!!";
            tvStatus.setText(aviso);
            Log.i(TAG,aviso);
        }

    }

    @Override
    public void deliveryComplete(IMqttDeliveryToken token) {
        Log.i(TAG, "msg delivered");
    }
});

/* Mqtt */
try {
    mqttAndroidClient.connect(mqttConnectOptions, null, new IMqttActionListener() {
        @Override
        public void onSuccess(IMqttToken asyncActionToken) {
            Log.i(TAG, "connect succed");
            tvStatus.setText("Conectado!");

            try {
                subscribeTopic(SUB_TOPIC_DIST);
                subscribeTopic(SUB_TOPIC_GEAR);
            } catch (MqttException e) {
                e.printStackTrace();
            }
        }

        @Override
        public void onFailure(IMqttToken asyncActionToken, Throwable exception) {
            Log.i(TAG, "connect failed");
```

```
                tvStatus.setText("Falha na conexão!");
            }
        });
    } catch (MqttException e) {
        e.printStackTrace();
    }

    Button pubButton = findViewById(R.id.publish);
    pubButton.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            publishMessage("alimentar");
            tvStatus.setText("Comando para alimentar o pet enviado:");
        }
    });
}

/** @param topic mqtt */
public void subscribeTopic(String topic) throws MqttException {
    try {
        mqttAndroidClient.subscribe(topic, 0, null, new IMqttActionListener() {
            @Override
            public void onSuccess(IMqttToken asyncActionToken) {
                Log.i(TAG, "subscribed succeed");
            }

            @Override
            public void onFailure(IMqttToken asyncActionToken, Throwable exception) {
                Log.i(TAG, "subscribed failed");
            }
        });

    }catch (MqttException e){
        e.printStackTrace();
    }
}

/** @param payload */
public void publishMessage(String payload){
    try{
        if(mqttAndroidClient.isConnected()==false){
            mqttAndroidClient.connect();
        }

        MqttMessage message = new MqttMessage();
        message.setPayload(payload.getBytes());
        message.setQos(0);
        mqttAndroidClient.publish(PUB_TOPIC, message, null, new IMqttActionListener() {
            @Override
            public void onSuccess(IMqttToken asyncActionToken) {
```

```
                Log.i(TAG, "publish succeed!");
            }

            @Override
            public void onFailure(IMqttToken asyncActionToken, Throwable exception) {
                Log.i(TAG, "publish failed");
            }
        });
    } catch (MqttException e){
        Log.e(TAG, e.toString());
        e.printStackTrace();
    }
  }
}
```