Modernização de Equipamentos Tradicionais para a Internet das Coisas

Gabriel Z. da Silva[1], Dra. Marli F. G. Hernandez[1], Dr. Marcelo S. P. Pessoa[2], Dra. Marbilia P. Sergio[3]

gabrielzanchetim@gmail.com, marlih@unicamp.br,

mpessoa@usp.br, marbilia.sergio@cti.gov.br

[1] Faculdade de Tecnologia (FT)
UNICAMP – da Universidade Estadual de Campinas

[2]Escola Politécnica (POLI)
USP – da Universidade de São Paulo

[3]Coordenação Geral de Projetos e Serviços (CGPS)
CTI/MCTI Renato Archer – Campinas/SP

Abstract. *This study explores the integration of traditional equipment into the Internet of Things (IoT), focusing on an ozone generator as a case study. Initially, the React Native mobile app development environment was selected based on academic articles and ISO 25000 standards. The project developed a mobile application that, together with an Arduino simulator, controls the generator via Bluetooth, demonstrating the feasibility of the proposal. The next step is to adapt the actual equipment for Bluetooth communication. This research confirms that the integration of traditional devices into IoT systems can be replicated in other equipment, offering new possibilities for interaction and functionality.*

Resumo. *Este trabalho investiga a integração de equipamentos antigos na Internet das Coisas (IoT), com um estudo de caso em um gerador de ozônio. Inicialmente, foi selecionado o ambiente de desenvolvimento de aplicativos móveis React Native, após avaliação baseada em artigos acadêmicos e na norma ISO 25000. O projeto desenvolveu um aplicativo móvel que, junto a um simulador feito em Arduino, controla o gerador via Bluetooth, demonstrando a viabilidade da proposta. A próxima etapa é adaptar o próprio equipamento para comunicação Bluetooth. Este estudo confirma que a integração de dispositivos tradicionais em sistemas IoT pode ser replicada em outros equipamentos, oferecendo novas possibilidades de interação e funcionalidade.*

1. Introdução

A Internet das Coisas – IoT – é um ambiente que permite a integração de ”todas as coisas”em uma rede única através de mecanismos de comunicação para que se possa otimizar o uso de equipamentos, facilitar a realização de comandos e supervisionar o seu funcionamento. Este projeto de pesquisa tem por objetivo desenvolver uma solução para que equipamentos eletrônicos tradicionais, que não foram concebidos para a utilização dessa nova tecnologia, possam ser incorporados à rede IoT. Será realizada uma pesquisa bibliográfica visando identificar as soluções existentes no ambiente acadêmico bem como

artigos tecnológicos que abordem o tema estudado. Como estudo de caso, será utilizado um equipamento que a equipe de pesquisa tenha acesso a fazer alterações e incorporar nele uma placa de comunicação com o celular para que possa ser desenvolvido um aplicativo para realizar os comandos de forma remota. Como resultados espera-se obter um roteiro para a realização da integração de equipamentos, bem como a construção de aplicativo para celulares.

2. Escolha de um ambiente de desenvolvimento para celulares visando a criação de aplicativo para IoT

O objetivo geral da pesquisa é fazer o desenvolvimento de um aplicativo para celulares visando integrar equipamentos eletrônicos tradicionais ao ambiente IoT. Para tanto é necessário escolher as ferramentas de desenvolvimento para celulares. De acordo com o [StatCounter 2023], um site de análise de tráfego da Web, 99,32% do mercado utiliza celulares com os sistemas operacionais Android, desenvolvido pela Google, e iOS, desenvolvido pela Apple. A escolha, portanto, fica entre utilizar as ferramentas desses fornecedores ou um ambiente multiplataforma que cria os aplicativos para ambos. Foi realizada uma pesquisa procurando identificar qual a solução mais adequada para esta situação. Uma busca na internet identificou no [SourceForge 2023] mais de 10 frameworks para desenvolvimento de aplicativos, muitos deles desconhecidos, incluindo alguns nomes populares como: Flutter, React Native, Cordova, Ionic e Xamarin. De acordo com [Vailshery 2023], um pesquisador de eletrônica de consumo, uma pesquisa realizada pelo JetBrains apresenta na figura 1 os frameworks mais utilizados entre 2019 e 2022. Também pode ser observado que Flutter, React Native, Xamarin e Unity são os mais utilizados.

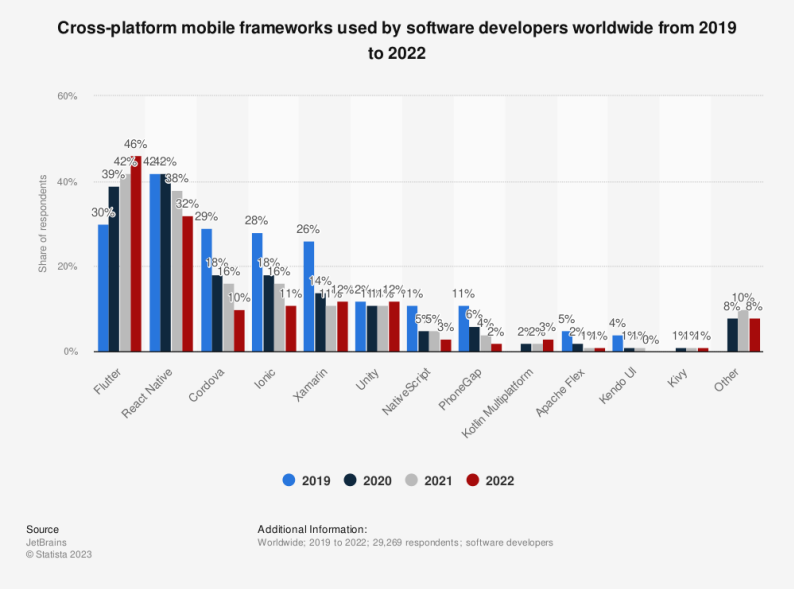

Figura 1. Frameworks multiplataformas utilizados pelos programadores de software mundialmente de 2019 a 2022. Fonte: [Vailshery 2023].

2.1. Artigos acadêmicos que analisam frameworks multiplataforma

[Hjort 2020] comparou Flutter e React Native, destacando o Flutter por seu ambiente de desenvolvimento robusto e desempenho rápido, enquanto React Native foi va-

lorizado por sua viabilidade a longo prazo e uso de APIs de renderização nativas. [Nawrocki et al. 2021] avaliou Flutter, React Native e Xamarin em um projeto prático, onde Flutter se destacou no desempenho e experiência de desenvolvimento, React Native ofereceu eficiência de memória e Xamarin mostrou-se inferior em vários aspectos, incluindo tempo de inicialização e falta de ”hot reload”. [Gülcüoğlu et al. 2021] analisou mais detalhadamente Flutter e React Native, sem identificar um vencedor claro; React Native foi elogiado pela facilidade de uso para desenvolvedores JavaScript e Flutter pela rapidez e menor dependência de bibliotecas externas.

2.2. ISO 25010 - Qualidade de produto de software

A escolha de um framework para desenvolvimento de aplicativos requer também a avaliação da qualidade desse ambiente e do produto que será gerado. A ISO possui uma série de normas, denominada ISO 25000 que tratam dos fatores da qualidade de produtos de software. [ISO 2022]

A ISO 25010 - Modelos de qualidade de sistema e software - pode ser usada para avaliação da qualidade de um produto de software. Esse modelo determina quais características da qualidade serão consideradas ao avaliar as propriedades de um software.

A qualidade de um sistema é o grau com que ele atende às necessidades declaradas de seus interessados, e, assim, proporciona valor. A qualidade do produto é categorizada em características e sub características. São elas:

- Adequação funcional: Integralidade funcional, Correção funcional e Adequação funcional.
- Eficiência de desempenho: Comportamento temporal, Utilização de recursos e Capacidade.
- Compatibilidade: Coexistência e Interoperabilidade.
- Usabilidade: Reconhecimento de adequação, Capacidade de aprendizagem, Operabilidade, Proteção contra erros do utilizador, Estética da interface do utilizador e Acessibilidade.
- Confiabilidade: Maturidade, Disponibilidade, Tolerância a Falhas e Recuperabilidade.
- Segurança: Confidencialidade, Integridade, Irrefutabilidade, Responsabilidade e Autenticidade.
- Manutenibilidade: Modularidade, Reutilização, Analisabilidade, Modificabilidade e Testabilidade.
- Portabilidade: Adaptabilidade, Instalabilidade e Substituibilidade.

A norma descreve mais detalhes sobre cada um dos tópicos relacionados e, para a escolha do framework, serão utilizados alguns desses itens como critério de avaliação.

2.3. Definição dos requisitos

Visando fazer uma ponderação mais objetiva para a escolha do ambiente de desenvolvimento, foram criados requisitos para permitir a comparação entre os diversos ambientes. Esses requisitos foram retirados de itens da Norma ISO 25010 e da análise dos artigos que apresentam esses ambientes. Foram organizados em:

- Requisitos para o desenvolvedor: Licença e custo; Plataformas suportadas; Distribuição; Viabilidade a longo prazo; Ambiente de desenvolvimento; Tempo de preparação; Manutenção;

- Requisitos para o usuário: Acesso ao hardware do dispositivo; Acesso à funcionalidade da plataforma; Internacionalização; Aparência e sensação; Desempenho.

2.4. Ponderação dos requisitos

- Licença e custo: 3. Uma vez que o projeto preza pelo uso de software livre, a licença e o custo associado a cada plataforma são importantes, mas não o fator principal.
- Plataformas suportadas: 5. Essencial, pois a aplicação deve ser compatível com várias plataformas de dispositivos móveis.
- Distribuição: 4. Importante, uma vez que a distribuição da aplicação é relevante para a acessibilidade dos usuários.
- Viabilidade a longo prazo: 4. Importante, pois a solução desenvolvida deve ser sustentável e duradoura no ambiente IoT.
- Ambiente de desenvolvimento: 2. Menos importante neste contexto, pois o foco está na integração com dispositivos IoT.
- Tempo de preparação: 4. Relevante para garantir que o desenvolvimento não seja excessivamente demorado, pois há um prazo determinado para a conclusão do projeto.
- Manutenção: 3. Importante, pois a manutenção da solução é fundamental para garantir seu funcionamento contínuo.
- Acesso ao hardware do dispositivo: 5 Crítico, pois a capacidade de acessar e controlar o hardware é essencial para a integração com dispositivos IoT.
- Acesso à funcionalidade da plataforma: 4. Importante para garantir que a aplicação possa interagir eficazmente com a plataforma móvel.
- Internacionalização: 2. Relevante para atender a um público global, mas não é o principal foco do projeto.
- Aparência e sensação: 2. Menos importante neste contexto, uma vez que o foco está na funcionalidade e na integração com dispositivos IoT.
- Desempenho: 4. Importante, pois um desempenho eficiente é essencial para uma aplicação IoT responsiva.

Tabela 1. Comparação ponderada entre os frameworks multiplataforma. Fonte: Autoria própria.

ID	CRITÉRIO	PESO	REACT NATIVE	FLUTTER
D1	*Licença e custo:*	**3**	5	5
D2	*Plataformas suportadas:*	**5**	5	5
D3	*Distribuição:*	**4**	5	5
D4	*Viabilidade a longo prazo:*	**4**	4	2
D5	*Ambiente de desenvolvimento:*	**2**	3	4
D6	*Tempo de preparação:*	**4**	4	3
D7	*Manutenção:*	**3**	5	5
U1	*Acesso ao hardware do dispositivo:*	**5**	5	5
U2	*Acesso à funcionalidade da plataforma:*	**4**	5	4
U3	*Internacionalização:*	**2**	4	4
U4	*Aparência e sensação:*	**2**	5	3
U5	*Desempenho:*	**4**	2	4
NP	**NOTA PONDERADA**	**-**	***4,38***	***4,14***

2.5. Ambiente de desenvolvimento selecionado

Tendo em vista a peculiaridade de cada framework e analisando os pontos mais relevantes dentro do projeto atual, após a adoção dos critérios ponderados da demanda, o ambiente escolhido para o desenvolvimento do aplicativo foi o React Native.

3. Implementação de uma aplicação

O controle remoto de dispositivos eletrônicos por interfaces móveis é cada vez mais comum, melhorando a integração e acessibilidade das tecnologias no cotidiano. O dispositivo escolhido para este projeto é um gerador de ozônio, usado em locais como clínicas veterinárias e hospitais, que possui uma interface tradicional com botões e LEDs. A limitação deste modelo está na sua operação fixa, que não permite controle à distância, um problema para locais de difícil acesso.

Para modernizar o uso do gerador, adicionamos capacidade de controle remoto via Bluetooth com um módulo HC-06, permitindo que usuários gerenciem suas funções através de smartphones e alinhando-o com o conceito de IoT. Essa atualização necessitou uma revisão significativa do código e do firmware do microcontrolador, além do desenvolvimento de uma interface de usuário intuitiva no aplicativo móvel. Estes ajustes são essenciais para assegurar a comunicação estável e transformar o gerador de ozônio em uma solução moderna e eficaz.

Figura 2. Gerador de ozônio tradicional. Fonte: Autoria própria.

4. Prova de conceito da comunicação

Neste capítulo será descrita como foi desenvolvida a prova de conceito de um sistema de comunicação com o celular via bluetooth. O trabalho será realizado em duas etapas: primeiramente um protótipo da comunicação com um Arduino que simula o funcionamento do equipamento de ozônio. A segunda etapa é alterar o equipamento de ozônio para receber essa comunicação.

4.1. Especificação do simulador de hardware

Devido à complexidade do sistema original do gerador de ozônio, optamos por projetar um protótipo que replicará as funcionalidades chave da máquina, utilizando um Arduino

UNO. Este protótipo simulará o comportamento dos LEDs e botões da máquina original e incorporará a conectividade Bluetooth através do módulo HC-06, permitindo o controle remoto. O Arduino será programado para manter a estrutura de estados do software original, enumerados de 1 a 5, respectivamente: Leitura de EEPROM; Repouso; Nível; Tempo e Operação.

4.2. Desenvolvimento de um aplicativo móvel

Um aplicativo móvel será desenvolvido utilizando React Native para permitir o controle remoto do protótipo. Os motivos para escolha dessa tecnologia já foram explicados anteriormente, e o aplicativo incluirá as seguintes funcionalidades: Interface de Usuário; Controle; Conectividade Bluetooth e Monitoramento.

5. Projeto e construção do simulador de hardware

5.1. Criação do protótipo

5.1.1. Utilização do Tinkercad

Para o desenvolvimento do nosso protótipo, utilizamos o Tinkercad, um aplicativo Web gratuito para projetos 3D, eletrônica e codificação [AutoDesk 2024]. O Tinkercad permite criar e testar circuitos eletrônicos em um ambiente virtual, facilitando a visualização e a experimentação sem a necessidade de componentes físicos. Essa abordagem é particularmente útil para validar a lógica de controle e a integração de componentes eletrônicos antes de proceder com a montagem real do hardware. No contexto do nosso projeto, o Tinkercad foi utilizado para simular o funcionamento do protótipo do gerador de ozônio, incluindo a interação entre o Arduino UNO e o módulo Bluetooth HC-06, além da lógica de controle dos LEDs e botões.

5.1.2. Integração do HC-06 e comunicação serial

No protótipo simulado, o módulo Bluetooth HC-06 foi conectado aos pinos RX (receptor) e TX (transmissor) do Arduino UNO. Essa configuração permite que o Arduino comunique-se com o HC-06 usando comunicação serial. A comunicação serial envolve o envio e recebimento de dados em formato binário, um bit por vez, através dos pinos TX e RX. No nosso sistema, essa conexão serial é utilizada para transmitir comandos do aplicativo móvel, via Bluetooth, para o Arduino, e vice-versa. Isso permite que o controle do dispositivo seja feito remotamente, estendendo a funcionalidade do gerador de ozônio para suportar operações via smartphone.

5.1.3. Simplificação do indicador LED

Devido às limitações de pinos disponíveis no Arduino UNO, fizemos uma adaptação no design original do sistema de LEDs. No sistema original, havia dois LEDs distintos, um para indicar ”operando”e outro para ”energia”. No protótipo desenvolvido, essas funções foram combinadas em um único LED. Esse LED tem comportamentos distintos para indicar diferentes estados do sistema:

- Aceso continuamente: Indica que o sistema está energizado e pronto para operar.
- Piscando: Significa que o sistema está ativamente operando.
- Desligado: Indica que o sistema não está recebendo energia.

5.1.4. Descrição detalhada do protótipo

Para criar um protótipo fiel às funcionalidades do gerador de ozônio original, enquanto incorporamos as capacidades de controle via Bluetooth, organizamos o layout e as conexões do Arduino UNO da maneira como é mostrada através da figura 3, que representa a disposição física dos elementos e as conexões.

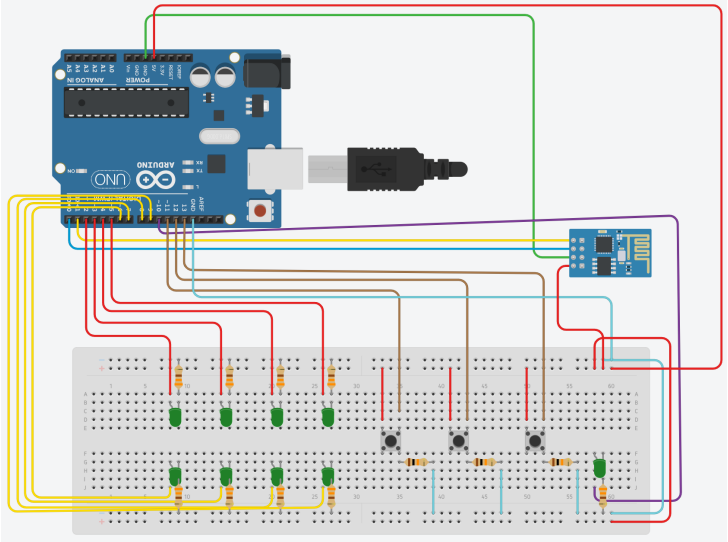

Figura 3. Circuito do protótipo desenvolvido no Tinkercad. Fonte: Autoria própria.

5.2. Implementação do código para Arduino

Para implementar o controle do protótipo do gerador de ozônio via Arduino, desenvolvemos um código que integra múltiplas funções, tais como comunicação via Bluetooth, controle de LEDs, leitura de botões e gerenciamento de estados operacionais. Este código foi adaptado a partir do software original do gerador de ozônio, modificando e expandindo as funcionalidades existentes para suportar a interação via aplicativo móvel. A seguir, descrevemos a lógica fundamental e a organização do código.

5.2.1. Diagrama de sequência do sistema

Antes de detalhar o código propriamente dito, é crucial entender a interação entre os componentes do sistema, que inclui o usuário, o aplicativo móvel, e o Arduino. Para isso, criamos um diagrama de sequência que ilustra como as operações são realizadas desde a abertura do aplicativo até a desconexão do dispositivo. Este diagrama nos ajuda a visualizar o fluxo de operações e a comunicação entre o aplicativo e o Arduino via Bluetooth, garantindo que todas as funcionalidades estejam sincronizadas e que o sistema responda de maneira eficaz aos comandos do usuário.

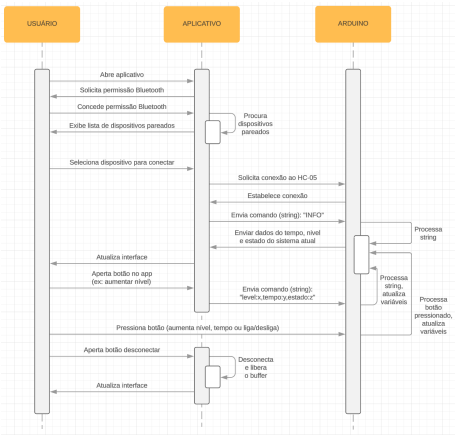

Figura 4. Diagrama de sequência do funcionamento do sistema. Fonte: Autoria própria.

5.2.2. Estrutura e funcionalidades do código do Arduino

O código para o Arduino foi estruturado para permitir comunicação serial com o módulo Bluetooth HC-06, leitura e escrita na EEPROM para manutenção dos estados entre reinicializações e controle de interfaces de usuário através de LEDs e botões. A maior parte das referências utilizadas vieram da documentação oficial do [Arduino 2024]. E as principais funcionalidades incluem: Inicialização e configuração; Comunicação serial com Bluetooth; Controle de estados; Feedback visual através de LEDs e Gerenciamento de eventos.

5.3. Criação do aplicativo do celular

Para o controle remoto do protótipo Arduino, desenvolvemos um aplicativo móvel utilizando React Native. Este aplicativo oferece uma interface de usuário gráfica que permite a interação com o Arduino via Bluetooth, facilitando o controle de configurações como níveis de operação e tempos de geração de ozônio. O desenvolvimento seguiu exemplos adaptados do exemplo feito por [Xukyo 2024], no website AranaCorp, que forneceu uma base para a comunicação Bluetooth. Além disso, utilizamos extensivamente a documentação oficial do [React 2024], que serviu como a principal fonte de referência técnica, garantindo a adoção das melhores práticas e das mais recentes funcionalidades disponíveis.

5.3.1. Estrutura e funcionalidades do aplicativo

O aplicativo móvel foi projetado para interagir eficientemente com o hardware via Bluetooth, apoiado pela documentação oficial do React Native. As funcionalidades do aplicativo incluem: Verificação do estado do Bluetooth; Descoberta de dispositivos; Estabelecimento de conexão; Leitura e processamento de dados; Envio de comandos; Interface de usuário; Gestão de estados e controle de operação; Permissões e segurança.

5.4. Acesso aos códigos

Os códigos completos para o Arduino e para o aplicativo móvel estão hospedados em um repositório público no GitHub para facilitar a revisão, colaboração e trans-

parência. Interessados podem acessar através do link: `https://github.com/gabrielzanchetim/Controlador-IC`.

6. Resultados dos testes do simulador de hardware

6.1. Protótipo Arduino

A figura 5 mostra o protótipo final montado no Arduino. Com as seguintes características: Configuração dos pinos; Botões de controle; Estética e organização.

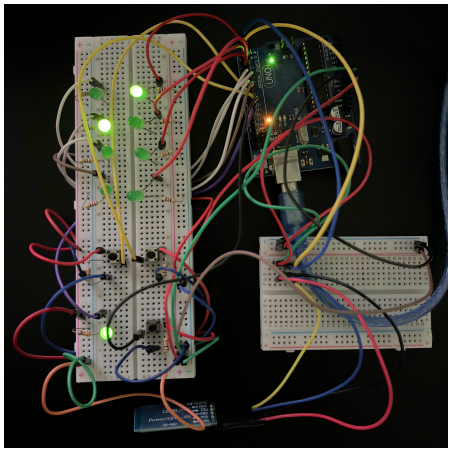

Figura 5. Protótipo na prática. Fonte: Autoria própria.

6.2. Aplicativo móvel

As capturas de tela a seguir mostram a interface do aplicativo desenvolvido, que permite a interação remota com o protótipo Arduino. As seguintes funcionalidades foram implementadas no aplicativo: Conexão Bluetooth; Controle de geração e tempo; Feedback visual e controle.

Figura 6. Tela de controle do aplicativo. Fonte: Autoria própria

6.3. Vídeo demonstrativo

Para ilustrar a integração completa e o funcionamento conjunto do protótipo Arduino com o aplicativo, disponibilizamos um vídeo demonstrativo no YouTube através do link: `https://youtu.be/852aHxDUuG8?si=f5vMDa_2HM0Y0BGN`.

7. Alterações no equipamento de ozônio

Demonstrado o bom funcionamento no simulador de hardware do equipamento, o próximo passo é alterar o equipamento para receber os comandos externos e apresentar o estado de funcionamento.

As alterações no projeto original implicam em alterações de hardware e software.

O hardware do PIC tem comunicação serial para se comunicar com a placa HC-06. Esses pinos estavam sendo utilizados para outras funções e é necessário um remanejamento para liberar essas portas para a função de comunicação serial. Felizmente algumas portas estavam sem uso e vão permitir essa alteração.

O software do PIC precisa ser alterado com três funcionalidades. A primeira é a comunicação serial com a placa bluetooth e o protocolo existente. A segunda funcionalidade é receber os comandos enviados pelo aplicativo do celular e a tercera é enviar para o celular o status de funcionamento.

Infelizmente não houve tempo hábil para implementar todas essas alterações no equipamento mas, com a implementação do simulador em arduino foi demonstrado que é perfeitamente possível realizar esta tarefa.

8. Considerações finais

Este trabalho apresentou o desenvolvimento de uma solução inovadora para integrar equipamentos eletrônicos ao ambiente de Internet das Coisas (IoT), utilizando um gerador de ozônio como estudo de caso. Através de pesquisas e análises de frameworks multiplataforma, optou-se pelo desenvolvimento de um aplicativo móvel em React Native. Este aplicativo não apenas facilita o controle remoto do equipamento via Bluetooth, mas também permite a coleta e armazenamento de dados na internet, ampliando a capacidade de comunicação com outros sistemas onde os dispositivos não só interagem localmente, mas também participam de uma rede global, integrando e sincronizando operações em diversas plataformas, que é o conceito de IoT. A criação de um protótipo utilizando o Arduino UNO demonstrou a viabilidade da solução proposta, permitindo uma interação intuitiva e eficaz. Os resultados alcançados confirmam que os objetivos deste projeto foram satisfatoriamente atingidos, oferecendo uma base sólida para futuras expansões e implementações em outros tipos de equipamentos eletrônicos.

9. Agradecimentos

O presente trabalho foi realizado com apoio do CNPq, Consellho Nacional de Desenvolvimento Científica e Tecnológico - Brasil e do CTI, Centro de Tecnologia da Informação Renato Archer.

Referências

Arduino (2024). Arduino documentation. Disponível em: https://docs.arduino.cc. Último acesso em: 04 de Agosto de 2024.

AutoDesk (2024). Autodesk tinkercad. Disponível em: https://www.tinkercad.com/dashboard. Último acesso em: 04 de Agosto de 2024.

Gülcüoğlu, E., Ustun, A., and Seyhan, N. (2021). Comparison of flutter and react native platforms. *Journal of Internet Applications and Management*.

Hjort, E. (2020). Evaluation of react native and flutter for cross-platform mobile application development. Master's thesis, Åbo Akademi University.

ISO (2022). Iso/iec 25010. Disponível em: https://iso25000.com/index.php/en/iso-25000-standards/iso-25010. Último acesso em: 11 de Novembro de 2023.

Nawrocki, P., Wrona, K., Marczak, M., and Sniezynski, B. (2021). A comparison of native and cross-platform frameworks for mobile applications. *Computer*, 54(3):18–27.

React (2024). React native. Disponível em: https://reactnative.dev/. Último acesso em: 04 de Agosto de 2024.

SourceForge (2023). Compare business software. Disponível em: https://sourceforge.net/software/. Último acesso em 25 de Outubro de 2023.

StatCounter (2023). Mobile operating system market share worldwide. Disponível em: https://gs.statcounter.com/os-market-share/mobile/worldwide/monthly-201809-202309. Último acesso em 25 de Outubro de 2023.

Vailshery, L. S. (2023). Cross-platform mobile frameworks used by software developers worldwide from 2019 to 2022. Disponível em: https://www.statista.com/statistics/869224/worldwide-software-developer-working-hours/. Último acesso em: 25 de Outubro de 2023.

Xukyo (2024). Criar uma aplicação bluetooth para o esp32 com react native. Disponível em: https://www.aranacorp.com/pt/criar-uma-aplicacao-bluetooth-para-o-esp32-com-react-native/. Último acesso em: 04 de Agosto de 2024.