

## Desenvolvimento de produtos IOT

### IOT products development

DOI:10.34117/bjdv7n1-320

Recebimento dos originais: 12/12/2020

Aceitação para publicação: 12/01/2021

#### **Ana Patrícia Fontes Magalhães Mascarenhas**

Doutora em Ciência da Computação

Instituição: Universidade Salvador

Endereço: Av. Luís Viana Filho nº 3100 / 3146. Imbuí, Salvador – Bahia

E-mail: ana.fontes@unifacs.br

#### **Sérgio Martins Fernandes**

Doutor em Engenharia da Computação pela USP

Instituição: Universidade Salvador

Endereço: Av. Luís Viana Filho nº 3100 / 3146. Imbuí, Salvador – Bahia

E-mail: sergiomfernandes63@gmail.com

#### **Fábio Duarte Freitas**

Estudante de Ciência da computação

Instituição: Universidade Salvador

Endereço: Av. Luís Viana Filho nº 3100 / 3146. Imbuí, Salvador – Bahia

E-mail: freitasfdf28@gmail.com

#### **Gabriel Borges Calheiros**

Estudante de Ciência da computação

Instituição: Universidade Salvador

Endereço: Av. Luís Viana Filho nº 3100 / 3146. Imbuí, Salvador – Bahia

E-mail: gabi\_borges01@hotmail.com

#### **Guilherme Luiz Garcia Lefrançois**

Estudante de Ciência da Computação

Instituição: Universidade Salvador

Endereço: Av. Luís Viana Filho nº 3100 / 3146. Imbuí, Salvador – Bahia

E-mail: gui.lefrancois21@gmail.com

#### **Marcela Braga Bahia**

Estudante de Ciência da Computação

Instituição: Universidade Salvador

Endereço: Av. Luís Viana Filho nº 3100 / 3146. Imbuí, Salvador – Bahia

E-mail: mahbraga0@gmail.com

#### **Victor Fernandes Baião Raton**

Estudante de Ciência da Computação

Instituição: Universidade Salvador

Endereço: Av. Luís Viana Filho nº 3100 / 3146. Imbuí, Salvador – Bahia

E-mail: vfbraton@gmail.com

## RESUMO

A Internet das Coisas (IoT) é uma rede de coisas físicas e aparelhos virtuais que se comunicam e interagem entre si. O desenvolvimento de produtos IoT tem se mostrado um desafio, pois demanda times multidisciplinares para lidar com hardware, software e integração entre eles, bem como requisitos complexos impostos pela heterogeneidade de plataformas e a segurança de dados, entre outras questões. Neste contexto, nosso trabalho investiga o desenvolvimento de produtos IoT e suas especificidades. Neste artigo apresentamos um produto IoT para monitoramento e controle de reservatórios de água. Os dados coletados ao longo do desenvolvimento deste produto serão utilizados futuramente para especificação de um processo de desenvolvimento de produtos IoT que deverá auxiliar desenvolvedores. O produto foi implementado em uma casa fictícia, em escala reduzida e experimental, e apresentou resultados satisfatórios quanto ao monitoramento e controle das caixas d'água realizadas à distância através de uma aplicação web.

**Palavras-chave:** Internet das Coisas, Desenvolvimento de Produtos, IoT.

## ABSTRACT

The Internet of Things (IoT) is a network of physical things and virtual devices that communicate and interact with each other. The development of IoT products has proved to be a challenge, as it requires multidisciplinary teams to deal with hardware, software and integration between them, as well as complex requirements imposed by the heterogeneity of platforms and data security, among other issues. In this context, our work investigates the development of IoT products and their specificities. In this article we present an IoT product for monitoring and controlling water reservoirs. The data collected during the development of this product will be used in the future to specify an IoT product development process that should assist developers. The product was implemented in a fictitious house, on a small and experimental scale, and presented satisfactory results regarding the monitoring and control of water tanks carried out remotely through a web application.

**Keywords:** Internet of Things, Product Development, IoT.

## 1 INTRODUÇÃO

A Internet das Coisas (IoT) é a rede de coisas físicas e aparelhos virtuais que se comunicam e interagem entre si (Debasis & Jaydip, 2011). O termo "coisas" refere-se a uma ampla gama de dispositivos como veículos, dispositivos vestíveis, sensores físicos, virtuais entre outros. Devido ao crescente número de dispositivos e a complexidade dos sistemas, a IoT enfrenta desafios, tais como (Ma, 2011): (i) heterogeneidade, (ii) disponibilidade de dispositivos, (iii) grande quantidade de dados, (iv) segurança e privacidade entre outros, que aumentam a complexidade de desenvolvimento dos seus diversos produtos.

Usualmente soluções IoT envolvem a utilização de sensores acoplados às "coisas" que coletam dados para serem processados por software para tomar decisões que

interferem no ambiente monitorado, por exemplo, através de atuadores (Isi-TICS, 2017). Desta forma, produtos IoT especificamente envolvem equipes multidisciplinares, engenheiros mecânicos, eletricitas, de computação, entre outros, necessárias para construir o hardware e o software relacionado ao produto. Conhecer as etapas envolvidas neste processo, identificar as tarefas necessárias a serem executadas por cada um dos envolvidos na construção de um produto e os artefatos a serem produzidos é importante para construir produtos com qualidade e segurança bem como para aumentar a produtividade do desenvolvimento.

Este projeto investiga as etapas de construção de produtos IoT em direção à definição de um processo de desenvolvimento de produtos IoT. Durante o projeto foram desenvolvidos dois produtos IoT: um controle automático de presença de alunos em sala de aula e um sistema de monitoramento de tanques de água em residências. Ao longo do desenvolvimento desses produtos foram identificados os elementos relevantes para um processo, tais como fases, atividades, artefatos e papéis envolvidos. Este artigo apresenta o desenvolvimento do produto de monitoramento de tanques de água.

A metodologia utilizada englobou o estudo teórico sobre a Internet das Coisas, bem como as tecnologias que a permeiam, como as placas e seus acessórios, como por exemplo o ESP8266 (Benchhoff, 2020). Em seguida, houve o desenvolvimento de um produto prático, reforçando a presença de características IoT. Inicialmente foi realizado um levantamento de necessidades englobando hardware e requisitos de software. Com base nestas necessidades foi construído o projeto físico da casa, selecionado o hardware a ser utilizado e realizada a implementação do hardware envolvendo desde a montagem dos tanques e a instalação dos sensores e atuadores até a implementação de baixo nível. O passo seguinte consistiu no projeto e implementação do software, bem como da solução de comunicação hardware e software. Finalmente o produto foi testado.

A Seção 2 deste artigo introduz o referencial teórico necessário para o melhor entendimento do trabalho e a Seção 3 apresenta outros trabalhos que também abordam o desenvolvimento de produtos IoT; Em seguida, a Seção 4 apresenta nosso sistema IoT - MoCoCA (Monitoramento e Controle de Caixas de Água), inicia com uma visão geral do produto e segue com o desenvolvimento deste. A Seção 5, detalha o aplicativo web de monitoramento de todo sistema, que é focado, principalmente, na comodidade de uso para o usuário. A Seção 6 mostra o cenário de testes do protótipo; e a Seção 7, a conclusão do projeto e os trabalhos futuros.

## 2 REFERENCIAL TEÓRICO

Esta seção apresenta algumas tecnologias envolvidas no desenvolvimento de produtos IoT que foram utilizadas no projeto descrito neste artigo.

Muitos produtos IoT utilizam como hardware as placas Arduino (Embarcados (ames) Arduino UNO, 2019), (Blog, 2019) por serem simples e de baixo custo. Essas placas estão em geral conectadas a sensores e atuadores acoplados à “coisa” que será conectada à rede. Sensores são dispositivos que respondem a estímulos, tais como luz e calor. Um estímulo capturado por um sensor é utilizado como entrada de dados em um sistema que os processa e como resultado pode interferir no ambiente através de atuadores, como, por exemplo, um motor. Desta forma, através dos dados coletados pelos sensores a placa Arduino pode se comunicar com um sistema computacional que fará o processamento desses dados e decidirá como atuar no ambiente. O Arduino tem um poder computacional relativamente baixo. Por isso é comum que se comunique com um servidor mais robusto capaz de realizar processamentos mais complexos.

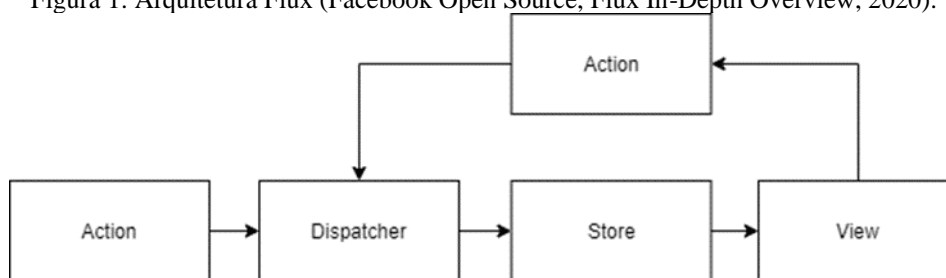
No contexto de software, aplicações elaboradas para IoT devem considerar alguns requisitos, tais como (Pierleoni et al, 2020): tolerância a falhas; reatividade, capacidade de reagir à eventos disparados para a aplicação; alto desempenho, uma vez que tendem a trabalhar, de modo constante, com questões de monitoramento em tempo real; e acessibilidade em múltiplas plataformas pois há uma gama de plataformas disponíveis em termos de dispositivos, tais como, *desktops*, *smart TVs*, *smartphones* e dispositivos vestíveis.

Arquiteturas cliente-servidor são comumente usadas na implementação de software neste contexto. Um exemplo de tecnologia que implementa esse cenário é a PWA (*Progressive Web Application*), que funciona como uma junção entre aplicação mobile e web. PWA é um modelo de implementação de arquitetura que usa como base o que é trabalhado em uma aplicação exclusivamente *web*, porém, com objetivo de ser implementado, também, em outros contextos de aplicações não relacionadas a web, como aplicativos nativos de uma determinada plataforma. Os principais benefícios de uma implementação PWA são: abstração de plataforma, devido a estrutura tender a se comportar de forma nativa. A abstração de funcionalidades nativas favorece o funcionamento multiplataforma, eliminando assim algumas inconveniências como processos de instalação e problemas de hardware; e funcionamento *off-line*, devido a implementação de *Service Workers*<sup>12</sup> e de telas estáticas. Uma PWA diferente de uma aplicação *web* não precisa estar conectada para estar funcionando.

A PWA é produzida utilizando a biblioteca React.js (React, 2020) para a criação dos componentes de interface com usuário (UI – User Interface). Construída pelo Facebook, a biblioteca React provê escrita de código componentizado com sintaxe JSX, uma fusão entre HTML e JavaScript (W3School, 2020). Os componentes, por sua vez, são quaisquer elementos que constituam a UI, como por exemplo botões e outros elementos interativos. Além da modularização de código, o React.js também contém uma árvore de elementos virtuais, a chamada Virtual DOM. A partir dela, elementos podem ser renderizados, mas de forma mais rápida que no DOM real provido pelo *Browser* por conter somente os elementos necessários para o funcionamento da aplicação em sua estrutura, desta forma, mudanças em elementos podem ser realizadas na árvore virtual e só então refletidas na árvore real.

A tecnologia PWA é implementada na arquitetura Flux que fornece a possibilidade de reação a eventos externos (eventos disparados de sistemas externos à aplicação) e internos (eventos disparados pela própria aplicação cliente, entre seus próprios componentes). A arquitetura Flux (Figura 1) é constituída das camadas: *View*, onde se situam os componentes visuais, fazendo interface com o usuário; *Action*, que contém o conjunto de ações e métodos que enviam informações à camada *Dispatcher*; *Dispatcher*, camada transmissora que decide que dado irá para cada *Store*; e *Store*, que contém a lógica de negócio e estado da aplicação.

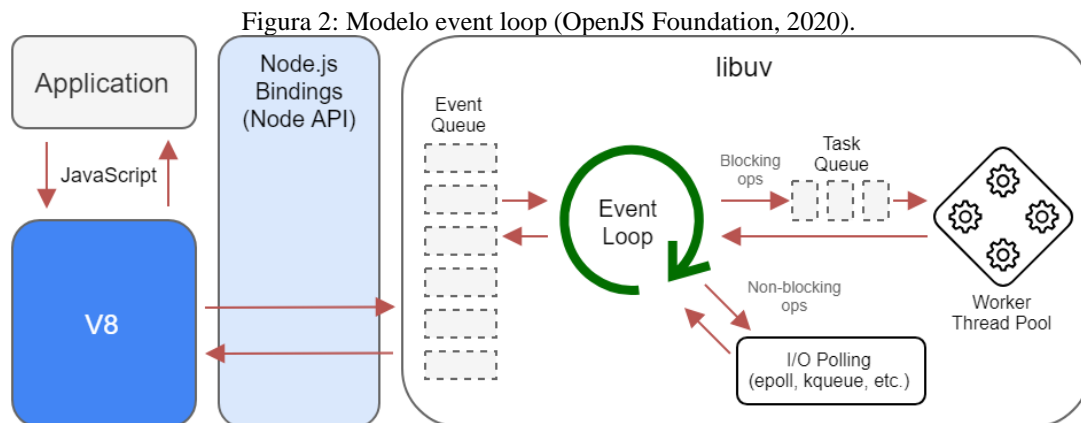
Figura 1: Arquitetura Flux (Facebook Open Source, Flux In-Depth Overview, 2020).



Outra tecnologia que também pode ser utilizada no contexto de IoT é o Node.js. Trata-se de uma runtime javascript que foi inicialmente desenvolvida por Ryan Dahl em 2009 com o intuito de trabalhar com implementações de entrada e saída (I/O) mas em vez de utilizar múltiplas *threads*, como as soluções mais comuns, é implementado em *single thread* visando utilizar os recursos computacionais de forma mais eficiente evitando ociosidade. De forma resumida o Node.js implementa o conceito de event loop para que os processos de I/O não sejam bloqueados, ou seja para que uma tarefa mais complexa não impeça as demais tarefas de serem executadas. Isso é feito por meio do *Event Loop*,

uma implementação na qual delega uma determinada tarefa ou requisição para uma *thread* e as requisições correm de forma assíncrona.

O modelo do *event loop* pode ser visto na imagem abaixo na Figura 2:



### 3 TRABALHOS RELACIONADOS

As etapas do desenvolvimento de um projeto IoT concentram-se no desenvolvimento dos elementos de hardware, estabelecimento da rede para conexão e desenvolvimento de software para a aplicação final. Empresas com a experiência no desenvolvimento de produtos físicos, que envolvem a produção de protótipos, sabem a necessidade de um refinamento contínuo com base no feedback de versões iniciais. A lista de perguntas pode ser grande e a resposta para cada uma delas estará nas etapas do desenvolvimento (ISI-TICs, 2017).

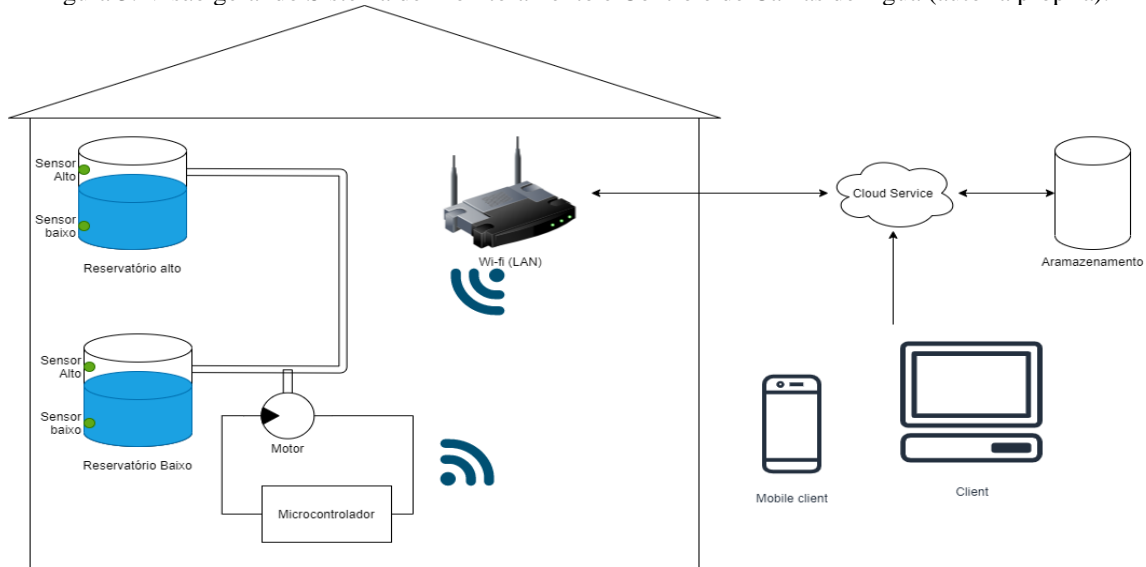
Já existem pesquisas que abordam o desenvolvimento IoT, como por exemplo: Interoperabilidade entre produtos (Arne et al, 2017), Composição de serviços para formar novos serviços (Davide et al, 2015), Linguagens de modelagem para produtos IoT (HyunJae et al, 2017), Geradores de código (Patel e Cassou, 2015), Modelagem de produtos com alto nível de abstração (Patel e Cassou, 2015), (Armin et al, 2020).

Essas pesquisas ainda estão em estágios iniciais no que se refere a um processo de desenvolvimento que oriente os desenvolvedores. Neste trabalho pretendemos obter informações em direção à especificação de um processo que atenda ao contexto de IoT.

### 4 SISTEMA IOT DE MONITORAMENTO E CONTROLE DE CAIXAS DE ÁGUA

Esta seção apresenta o Sistema de Monitoramento e Controle de Caixas de Água (MoCoCA) desenvolvido neste trabalho. A Figura 3 ilustra uma visão geral do produto.

Figura 3: Visão geral do Sistema de Monitoramento e Controle de Caixas de Água (autoria própria).



A residência ilustrada na figura contém dois reservatórios, um que fica no alto da casa, chamado de *reservatório alto*, e outro que fica na parte de baixo da casa, chamado de *reservatório baixo*. Em cada um deles existem dois sensores, um sensor de nível alto e outro sensor de nível baixo. Os reservatórios estão ligados através de um cano e neste está conectado um motor bomba, que por sua vez, tem acoplado em si um microcontrolador. Quando o nível de água do reservatório alto atinge o sensor baixo, um sinal é enviado ao microcontrolador, este sinal envia os dados dos reservatórios para a aplicação, cabendo ao usuário decidir ou não se acionar o motor bomba.

Todo o sistema é monitorado e controlado via wifi por um sistema web que pode ser acessado por um celular ou por um computador qualquer em tempo real. Os dados coletados pelos sensores são também armazenados em um banco de dados hospedado na nuvem.

As subseções a seguir detalham o desenvolvimento do sistema.

#### 4.1 PROJETO DE HARDWARE

A tabela a seguir apresenta os hardwares adotados, isto é, sensores, atuadores. As seguintes peças foram utilizadas:

Tabela 1: Hardware utilizado no projeto físico do sistema de monitoramento e controle de caixas d'água.

Nome	Quantidade	Valor	Porta	Ação
Módulo ESP8266	1	R\$ 30,00	--	Controlador, comunicação do WI-FI e substituto do Arduino
Sensor de fluxo d'água	1	R\$ 12,00	12	Sinaliza a entrada de água e sua vazão no momento
Sensor de nível d'água	3	R\$ 7,00	4,12 e 14	Sinalizam os estados de completude das caixas d'água.
Relé	1	R\$ 0,50	7	Controla a bomba hidráulica da residência.

## 4.2 PROJETO DO SOFTWARE

O projeto do software se iniciou com o levantamento dos requisitos necessários. Dentre os requisitos funcionais estão “Ler nível da água no tanque”, “Calcular vazão.”, “Acionar motor”.

Dessa forma percebemos as principais necessidades que deveriam ser atendidas pela solução, assim nos levando a definir os seguintes requisitos não funcionais.

- **Escalabilidade**, para aumentar a capacidade de demanda de forma fácil, pois nessa solução para otimizar o custo benefício teremos que evitar ociosidade de recurso computacional. Dessa forma a solução precisa “escalar”. Quando necessário atender a um volume de demandas acima do comum de forma fácil e sem efeitos colaterais; ou quando o consumo de recursos estiver abaixo do disponível, desativar os recursos ociosos.
- **Suporte multiplataforma**, para atender a diferentes tipos de clientes, não apenas em termos de software ou hardware, mas em termos de contexto (ex. residências e estabelecimentos comerciais).
- **Alta performance**: para tratar uma quantidade massiva de dados que podem ser alterados em infinitas vezes dado um período de tempo.

De forma a atender tais requisitos, optou-se por uma implementação utilizando linguagem de programação Javascript, sendo usado Node.js no lado servidor (back-end) e React.js no lado cliente (front-end).

Quanto ao desenvolvimento da aplicação cliente, desenvolveu-se uma PWA (progressive Web Application) utilizando a biblioteca React.js. Essa decisão está relacionada à flexibilidade, pois uma PWA, trata-se de um modelo de aplicação onde o artefato final trata-se de uma aplicação multiplataforma baseada na web, que consegue se integrar de forma nativa ao sistema operacional do cliente em questão garantindo flexibilidade e dispensando a necessidade de instaladores.

Para trabalhar a interoperabilidade entre dispositivos de diferentes contextos, estabeleceu-se um modelo de comunicação assíncrona por meio do protocolo de



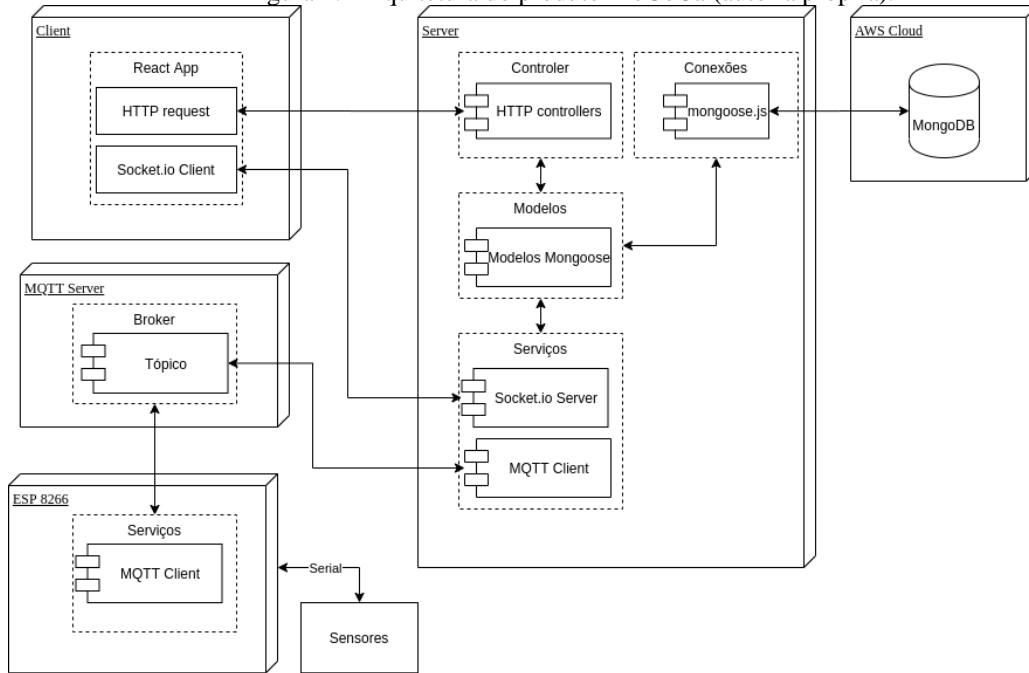
comunicação MQTT para permitir uma comunicação bidirecional entre api de serviço e microcontrolador e o uso do websocket para comunicar o serviço com os demais aplicativos clientes.

A implementação de tais soluções em um único projeto resultou em um serviço para controle de aplicações de IoT genérico cuja arquitetura pode ser representado por meio do diagrama de componentes apresentado na Figura 4.

Como pode ser observado na figura, o componente central *Server* contém os sub componentes *Controler*, *Conexões*, Modelos e serviços. Na nuvem (*ADS Clowd*) fica hospedado o banco de dados em MongoDB. Observa-se também o Client, onde executará a aplicação cliente, e a parte física do projeto, representada na figura como o ESP8266 e os sensores. A comunicação é provida pelo componente MQTT Server.

Conforme ilustrado na figura, para o armazenamento das informações optou se pela implementação da solução NoSQL MongoDB que nos oferece um suporte mais adequado a nossa solução, pois diferente de uma implementação SQL, o Mongoddb não usa sistema de tabelas e sim de coleções que possuem funções padrões de inclusão, alteração, exclusão e busca (com filtro), além de maneira de estimar o tamanho dessas coleções(*counts*). Além de oferecer recursos de escalabilidade visto que o MongoDB pode ser aplicado como um banco de dado distribuído de forma simples, a solução NoSQL nos dá uma liberdade e flexibilidade em relação aos dados salvos de forma que temos uma maior liberdade de modelagem de nosso sistema de armazenamento.

Figura 4: Arquitetura do produto MoCoCa (autoria própria).



## 5 APLICATIVO DE MONITORAMENTO

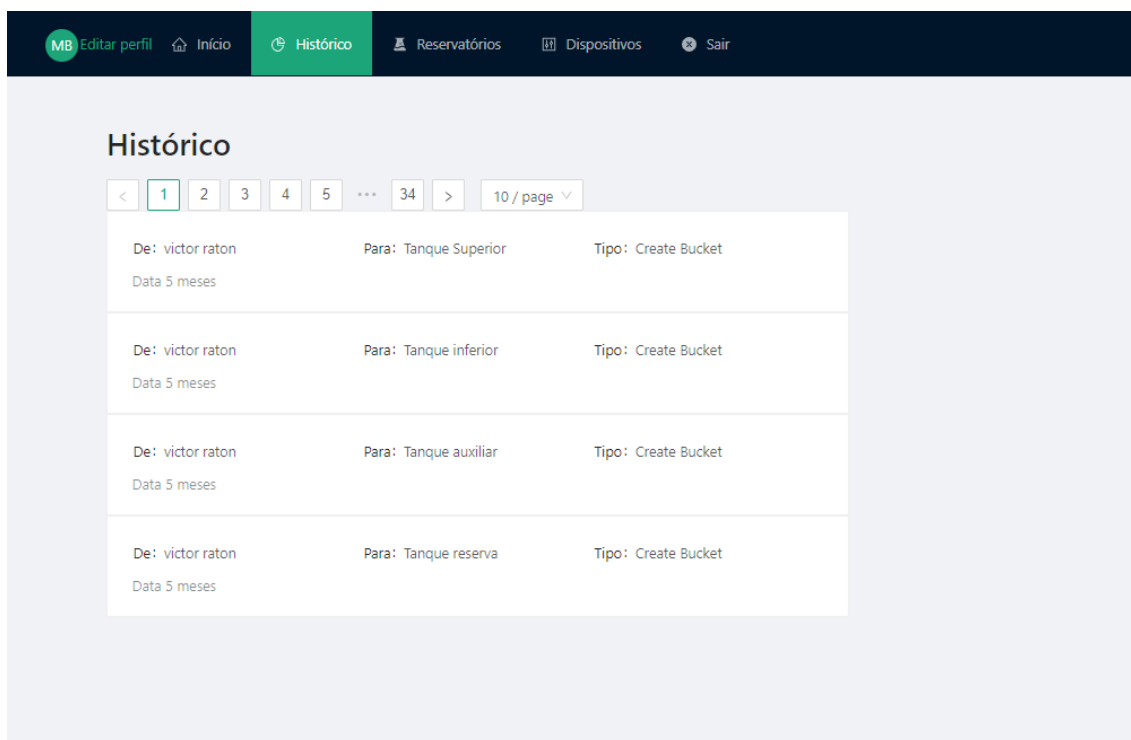
O nosso projeto contempla também um sistema Web concebido para suprir a necessidade do proprietário da residência de monitorar, à distância, os dados operacionais do sistema MoCoCa. A partir da aplicação produzida, se pode visualizar e controlar os dispositivos que compõem o sistema, como atuadores e sensores, bem como visualizar dados estatísticos de funcionamento.

O sistema foi criado em React.js e abriga diversas funcionalidades de gerenciamento do sistema. Nele, o usuário devidamente cadastrado passa por autenticação para que possa interagir com a aplicação, podendo, nessa, registrar novos reservatórios e dispositivos (atuadores, sensores, microcontroladores e outros) que existam fisicamente bem como visualizar os que já foram cadastrados previamente.

A Figura 5 apresenta a tela de histórico de manipulação de dispositivos no sistema. Nela, ocorre a descrição de elementos quaisquer como dispositivos, atuadores e sensores que são criados, excluídos e editados por um usuário, ao longo do tempo. Nesse caso, a figura mostra a criação de reservatórios d'água.

A principal funcionalidade do sistema é o monitoramento dos reservatórios. O cliente pode visualizar a situação dos reservatórios, isto é, o nível de água atual de cada um deles, além de acompanhar o funcionamento do sistema, se a bomba está ligada ou desligada e qual a vazão da água.

Figura 5: Tela de histórico da interface web.



De:	Para:	Tipo:
victor raton	Tanque Superior	Create Bucket
Data 5 meses		
victor raton	Tanque inferior	Create Bucket
Data 5 meses		
victor raton	Tanque auxiliar	Create Bucket
Data 5 meses		
victor raton	Tanque reserva	Create Bucket
Data 5 meses		

## 6 VALIDAÇÃO DO SISTEMA

O artefato resultante do processo de desenvolvimento trata-se de um MVP (Mínimo Produto Viável) com o fim de atestar a capacidade de implementar IoT para a resolução do problema apresentado, dessa forma, nota-se que seja uma solução minimizada.

Os testes foram divididos em três etapas: (i) teste físico; (ii) teste do aplicativo web; (iii) teste da solução completa.

O teste físico compreendeu a integração dos componentes físicos, tais como sensores, bombas e placas controladoras. O teste se iniciou com os tanques vazios. Em seguida o tanque inferior começou a ser enchido com água até atingir o limite dos sensores. Neste momento, o motor da bomba foi acionado automaticamente para encher o tanque superior. Ao atingir o limite do tanque superior (leitura do sensor) a bomba foi automaticamente desligada. Este teste foi realizado diversas vezes considerando diferentes velocidades de vazão da água.

Para o teste do aplicativo web alimentamos manualmente o banco de dados com diferentes combinações de leitura dos sensores e analisamos os resultados apresentados na aplicação. Também avaliamos os comandos de ligar/desligar a bomba.

O teste da solução completa compreendeu avaliar a comunicação da parte física com o banco de dados, para gravação das leituras dos sensores em tempo real, e a comunicação da aplicação web com o protótipo físico considerando os comandos de ligar/desligar as bombas.

Após alguns ajustes os resultados foram considerados satisfatórios, isto é, constatou-se que foi possível monitorar e controlar à distância os tanques de água do nosso protótipo de residência.

## **7 CONCLUSÃO E TRABALHOS FUTUROS**

Neste trabalho desenvolvemos o Sistema para Monitoramento e Controle de Caixas de Água (MoCoCa) com o objetivo de identificar etapas e tarefas envolvidas no desenvolvimento de produtos IoT em prol da definição de um processo que possa nortear desenvolvedores em futuras implementações.

Foram identificadas as etapas e tarefas principais, dentre elas o projeto de hardware e software, a definição de tecnologias que atendam aos requisitos de IoT e especificidades do desenvolvimento de software de controle. O sistema MoCoCA foi desenvolvido e testado em um protótipo feito em escala reduzida. Os testes foram satisfatórios, embora uma replicação em ambiente real ainda precise ser realizada.

Atualmente estamos especificando um processo de desenvolvimento que possa auxiliar os projetistas ao longo da construção de produtos IoT que será validado na construção de um novo produto.

## REFERÊNCIAS

- Blog, 2019. Disponível em: <<https://blog.arduino.cc/>>. Acesso em: 18 out. 2019.
- Benchoff, Brian (2020). The Current State of ESP8266 Development. Disponível em <https://hackaday.com/2014/09/06/the-current-state-of-esp8266-development/>, acessado em 21/12/2020.
- Arne, B., Stefan, S., Corina Kim, S., Abdelmajid, K., & Sebastian, K. (2017). Enabling IoT Ecosystems through Platform Interoperability. *IEEE Software*, 54-61.
- Debasis, B., & Jaydip, S. (2011). Internet of Things: Applications and Challenges. Springer Science+Business Media. *Wireless Pers Commun*, pp. 49-69.
- ISI-TICs (2017), Etapas do Desenvolvimento de uma Solução em IOT. Disponível em <https://isitics.com/2017/12/05/etapas-do-desenvolvimento-de-uma-solucao-em-iot>, acessado em 31/07/2020.
- Ma, D. "Internet of Things: Objectives and Scientific Challenges," *Journal of Computer Science and Technology*, pp. 919-924, 11 2011.
- Pierleoni, P.; Concetti, R.; Belli, A. and Palma, L. (2020). Amazon, Google and Microsoft Solutions for IoT: Architectures and a Performance Comparison. In *IEEE Access*, vol. 8, pp. 5455-5470, doi: 10.1109/ACCESS.2019.2961511, 2020.
- OpenJS Foundation (2020), The Node.js Event Loop, Timers, and process.nextTick(), Disponível em: <https://nodejs.org/en/docs/guides/event-loop-timers-and-nexttick/#:~:text=What%20is%20the%20Event%20Loop,the%20system%20kernel%20whenever%20possible> acessado em 26/05/2020.
- Facebook Open Source, Flux In-Depth Overview (2020), Disponível em: <https://facebook.github.io/flux/docs/in-depth-overview>, acessado em 23/04/2020.
- Embarcados (ames) ,Arduino UNO (2019), Disponível em: <https://www.embarcados.com.br/arduino-uno/>, acessado em 10/05/2019.
- React, A Java Script Library for Building User Interfaces. Disponível em <https://reactjs.org/>, Acessado em 23/12/2020.
- W3School site oficial. Disponível em <https://www.w3schools.com/>, Acessado em 23/12/2020.
- Arne, B.; Stefan, S.; Corina Kim, S.; Abdelmajid, K.; Sebastian, K. Enabling IoT Ecosystems through Platform Interoperability. *IEEE Software*, pp. 54-61, 2017.
- Davide, C. ; Paolo, B.; Prabhakaran, K.; Claudio, P. ; Ferry, P.; Cultrona, P. A. Industrial application development exploiting IoT vision and model driven programming. Em 18th International Conference on Intelligence in Next Generation Networks, 2015.

HyunJae, L.; Eunjin, J.; Donghyun, K.; Jinmyeong, K.; Soonhoi, H. A novel service-oriented platform for the internet of things. Em 7th International Conference on the Internet of Things, Austria, 2017.

Patel P.; Cassou, D. Enabling high-level application development for the Internet of Things. Journal of System and Software, 2015.