

# Construindo um Smart IoT Gateway

Danilo Guimarães \*

Ricardo Rodrigues †

Uillan Araújo ‡

15 de Setembro de 2017

## Resumo

De acordo com o grande avanço da Computação, torna-se comum a coexistência de objetos reais com o mundo virtual. Devido a essa evolução, têm-se a necessidade de uma padronização de conceitos como a *Internet of Things* (IoT) com a comunicação de seus dispositivos, assim surge a ideia de um Gateway para agregar e centralizar essa transição de informações. Este trabalho propõe um estudo breve sobre Gateway IoT, isto é, construir um software Smart IoT Gateway funcional e utilizável em projetos de pequeno e médio porte.

**Palavras-chaves:** Internet of Things. IoT Gateway. Arquitetura de Software.

## Abstract

According to the great advance of Computing, it becomes common the coexistence of real objects with the virtual world. Due to this evolution, there is a need for a standardization of concepts such as Internet of Things (IoT) with the communication of its devices, thus the idea of a Gateway to aggregate and centralize this transition of information. Thus, this work proposes a brief study on Gateway IoT, that is, to build an Smart IoT Gateway functional and usable in small and medium-sized projects.

**Key-words:** Internet of Things. IoT Gateway. Software Architecture.

## Introdução

Com os recentes avanços das tecnologias, especificamente nas últimas décadas e devido a democratização da Internet, nossa sociedade tem caminhado para um cenário cada vez mais conectado. Se antes apenas super-computadores e máquinas robustas eram conectadas à rede, a tendência nos próximos anos é que dispositivos cada vez menores também tenham seu espaço na Internet. A essa tendência chamamos *Internet of Things* (Internet das Coisas ou IoT). É uma nova visão que descreve objetos fazendo parte da rede,

---

\*guimaraesdjl@gmail.com

†ricardo.faria@outlook.com.br

‡uillan@outlook.com

onde cada um deles é unicamente identificado, acessível através da rede, com posição e estado conhecido, captando informações sensoriais ou agindo sobre o ambiente. Serviços são construídos com base nesses objetos <sup>1</sup>. Estima-se que até 2020, sejam investidos cerca de US\$ 267 bi na indústria e serviços voltados para IoT <sup>2, 3</sup>, ver Figura 1.

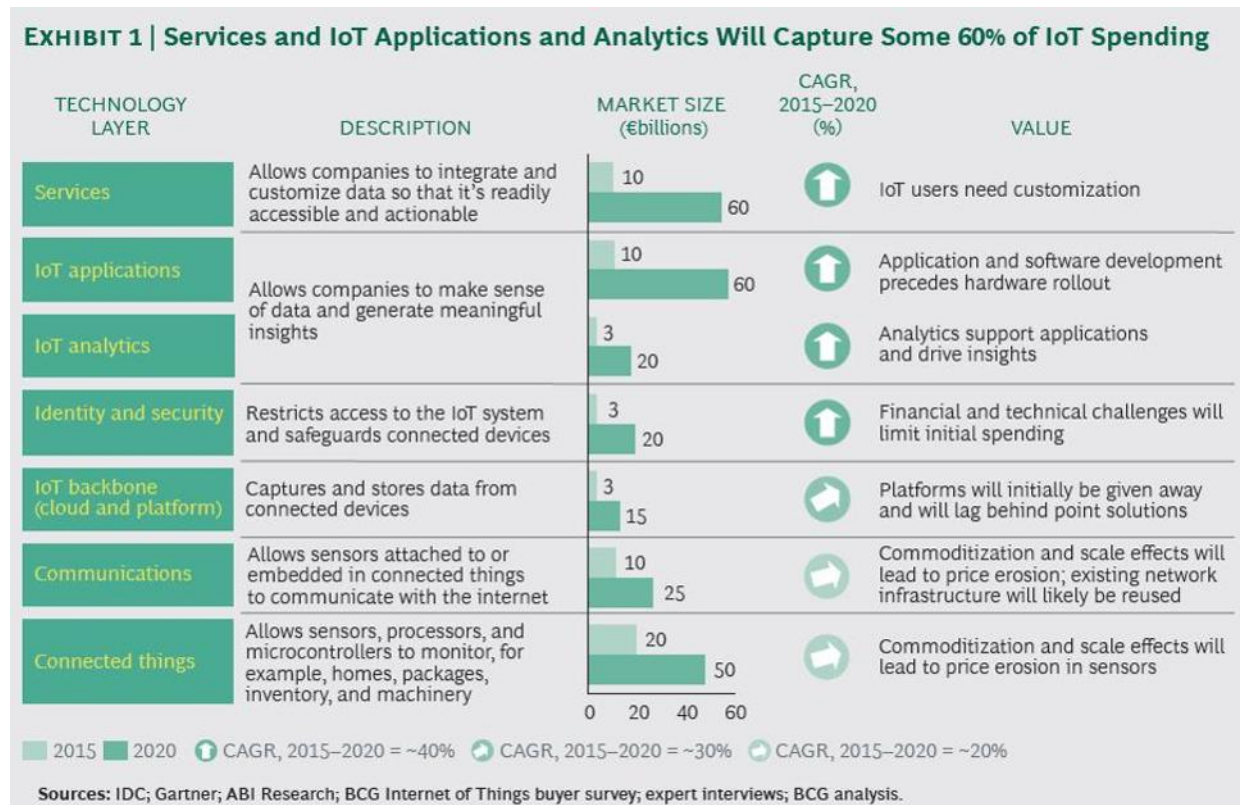


Figura 1 – Dados dos investimentos futuros a *IoT*.

As pressões comerciais são implacáveis. Como resultado disso, os requisitos estão se tornando cada vez mais complexos, exigindo assim uma maior conectividade entre dispositivos legados e novos, o que leva à necessidade de personalizações. À medida que a conectividade aumenta, novos pontos de ataque surgem, levando as empresas a ficarem cada vez mais preocupadas com percas provenientes de ataques cibernéticos. Devido a isso, as empresas enfrentam um desafio ao encontrar uma solução que atenda as suas necessidades específicas, e isso está impulsionando a necessidade de dispositivos IoT personalizados. Assim surgem os Smart IoT Gateway <sup>4</sup>.

Para entender melhor como as tecnologias envolvidas funcionam, o objetivo deste trabalho é construir um Smart IoT Gateway open-source, funcional e de uso simples, onde usuários cadastrem ações baseados nos dados enviados por um sensor cadastrado. Neste trabalho não temos como objetivo construir um projeto de hardware para um Gateway IoT.

## 1 Internet of Things

A Internet começou como uma forma do governo comunicar após uma guerra nuclear, mas evoluiu para ser muito mais do que uma rede. De muitas maneiras, a Internet

tornou-se um mundo digital que tem ligações ao nosso mundo físico <sup>5</sup>.

Conforme as tecnologias avançam, torna-se cada vez mais comum que todos estejam conectados. Deste modo, a evolução dos objetos físicos passam a coexistir com a Internet, impactando em diversos aspectos no cotidiano das pessoas, seja no profissional ou pessoal.

A Internet das Coisas (do inglês, *Internet of Things*, ou simplesmente IoT) é a revolução tecnológica que visa conectar dispositivos eletrônicos (como aparelhos eletrodomésticos, máquinas industriais, meios de transporte etc.) à Internet. IoT é um termo criado por Kevin Ashton <sup>6</sup>, um pioneiro da tecnologia britânico que concebeu, em 1999, um sistema de sensores onipresentes conectando o mundo físico à Internet, enquanto trabalhava em identificação por rádio frequência (RFID). O grande valor da IoT está no preenchimento das lacunas entre o mundo físico e digital em sistemas <sup>7</sup>.

Na sua essência, a IoT significa apenas um ambiente que reúne informações de vários dispositivos (computadores, smartphones, semáforos, e quaisquer objeto com um sensor) e de aplicações (qualquer coisa desde uma aplicação de mídia social como o Twitter à uma plataforma de comércio eletrônico, de um sistema de produção à um sistema de controle de tráfego). Quando se combinam informações de dispositivos e de outros sistemas, enormes recursos de processamento são utilizados para análises expansivas, geralmente associadas com o conceito de *Big Data* <sup>1</sup> – ou seja, a análise de dados não necessariamente concebidos para serem avaliados em conjunto. Esta noção de múltiplas finalidades é provavelmente a melhor razão para usar o termo “Internet das Coisas”, quando a Internet é mais do que uma rede resistente para ser um canal para qualquer combinação e coleção de atividades digitais <sup>1</sup>.

Em seu processo evolutivo a IoT enfrenta diversos problemas, que variam de aplicativos (sistemas), políticas de segurança e até problemas técnicos. Com todos estes dispositivos conectados à Internet uma enorme quantidade de informação é disponibilizada levantando questões de confiabilidade destas informações. Onde e quem garantirá a autenticidade dessas informações? Quem pode ter acesso à essas informações? Quem irá proteger essas informações? São alguns dos problemas enfrentados ao se disponibilizar as informações de objetos do mundo físico ao mundo virtual. Uma padronização entre as tecnologias se torna muito importante, pois ela levará a uma melhor interoperabilidade, reduzindo barreiras. Muitos fabricantes estão criando suas próprias soluções (Intel, Dell etc.) o que as levam à ter comportamentos diferentes, dificultando a integração destes sistemas ou dispositivos <sup>8, 9</sup>.

## 2 Gateway IoT

Ao utilizar IoT, imediatamente teremos algo que estará conectado à Internet. Esse “algo” não necessariamente se conecta de forma direta. Na grande maioria dos casos, essa conexão se dá por meio um gateway. Gateway IoT é uma aplicação (ou dispositivo com aplicação embarcada) responsável por receber requisições de diversos sensores e em algumas situações executar ações <sup>10</sup>.

O gateway é similar a um roteador <sup>2</sup>, porém, ele pode unir redes de diferentes protocolos através de um processamento local para a tradução e conversão de protocolos.

---

<sup>1</sup> *Big Data* é um termo usado para referenciar grandes e complexos volumes de informações que necessitam de técnicas e ferramentas específicas para serem capturadas, gerenciadas e/ou processadas

<sup>2</sup> Roteador é um dispositivo físico de rede cujo objetivo é encaminhar pacotes entre redes de computadores

Gateways são muito utilizados em ambiente industrial e corporativo, porém, com o avanço da IoT, é mais comum encontrar esse tipo de equipamento para uso residencial.

Com a evolução dos gateways são utilizados basicamente dois tipos de Gateways IoT<sup>11</sup>, os *Traditional Gateways* que não são inteligentes e apenas armazenam a informações para futura transmissão e os *Smart Gateways* que além disso, podem fazer ações, tais como:

- Persistir as informações;
- Efetuar transformação dos dados recebidos;
- Guardar os dados temporariamente para posteriormente transmiti-los para a Internet;
- Executar regras de segurança sobre os dispositivos;
- Executar ações com base nos dados recebidos e regras configuradas.

A Figura 2 ilustra uma arquitetura convencional de um Gateway IoT e suas relações com os sensores e com o Data Center.

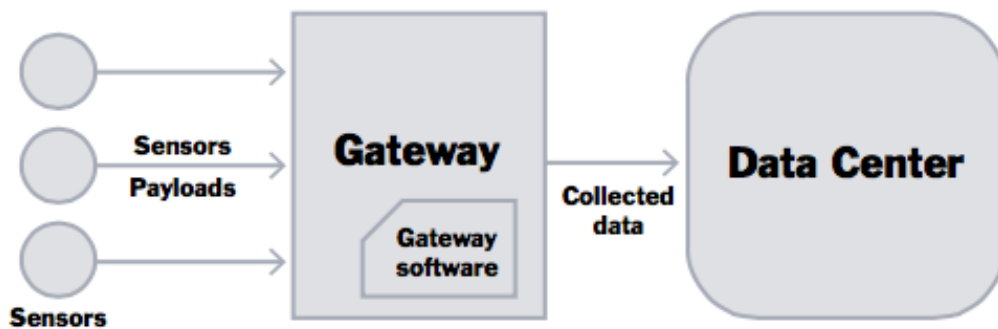


Figura 2 – Descrição esquemática da arquitetura básica de um Gateway IoT<sup>12</sup>. Os *Sensors* representam os *Sensores* e *Sensors Payloads* são as informações que cada um deles envia ao Gateway. O *Gateway Software* (objeto de estudo deste trabalho) é responsável por controlar o Gateway e o *Collected Data* é o conjunto de informações que o Gateway julgou oportuno enviar ao *Data Center*, que é o local onde os dados são armazenados e processados.

## 2.1 Sensores

Sensores são dispositivos, geralmente eletrônicos, que reagem a um determinado estímulo ou mudança no ambiente onde ele se encontra instalado. Seu principal propósito é o de enviar as informações para outros dispositivos presentes no sistema para que a informação seja processada. Em uma rede de sensores, tais dispositivos são instalados no que a indústria nomeia como sendo "campo", ou seja, onde os sensores devem coletar informações. São exemplos de campos: galpões, silos, plantas industriais, florestas, plantações etc. São exemplos de aspectos de ambiente em que sensores atuam: temperatura, umidade relativa do ar, altura, peso, pressão, ausência ou presença de um determinado objeto, etc. Já o Data Center tem o mesmo significado que estamos habituados: um servidor onde dados

são mantidos e processamentos são realizados. O Gateway IoT faz exatamente essa ligação entre os sensores e a Internet, uma vez que sensores não costumam ter boas condições de acesso à Internet, tanto do ponto de vista da disponibilidade quanto da largura de banda.

## 2.2 Vantagens

A grande vantagem em construir um Gateway open-source é o alto nível de customização que ele pode oferecer, já que temos total acesso ao sistema operacional, sem restrições impostas pelo fabricante – o que ocorre na maioria dos casos. Essa customização vai permitir usar o gateway em modo *fog computing*<sup>3</sup> para o processamento de informações o mais perto do dispositivo da borda, ou *edge device*, fazendo com que, mesmo na falta de Internet, o dispositivo consiga se manter operacional, ainda que com algumas restrições.

Fazer o seu próprio gateway de IoT pode parecer contra-intuitivo à primeira vista, mas com a baixa nos preços dos SoC's (System-on-a-Chip)<sup>4</sup>, alavancado principalmente pela Raspberry Pi<sup>5</sup>, torna possível a criação de sistemas computacionais de bom desempenho, tamanho reduzido e baixo custo. Já é possível encontrar SoC's custando menos de US\$ 10 e chips completos e funcionais por menos de US\$ 40<sup>14</sup>.

## 3 Trabalhos Relacionados

De acordo com as fontes pesquisadas, especificamente a biblioteca digital da IEEE, existem poucos trabalhos relacionados específicos à Smart IoT Gateway.

O primeiro artigo<sup>15</sup> analisado possui similaridades com a nossa solução de Smart IoT Gateway, ele apresenta um software Smart Iot Gateway flexível que pode ser configurado para se adaptar a diferentes requisitos, e com isso reduzir custos e facilitar o ciclo de desenvolvimento. O Smart IoT Gateway tem como responsabilidade controlar os diferentes protocolos de comunicação, identificar os tipos de dispositivos e possíveis tratamentos de mensagens. Além disso, o gateway possui algumas interfaces externas unificadas, capazes de se adaptar à diferentes necessidades, onde os usuários podem desenvolver cartões apropriados de acordo com diferentes aplicativos. Como Ethernet (RJ45), USB, som e interfaces de vídeo, ver Figura 3.

Já o segundo artigo<sup>16</sup> também apresenta algumas similaridades com a nossa proposta. Propõe um Smart IoT Gateway que atua como um elemento central para conectar dispositivos através de protocolos e tecnologias de comunicação: ZigBee, Wifi, Bluetooth e Ethernet. Garante também a implementação das funcionalidades de conversão de protocolos, transformação, processamento e armazenamento de dados. Um dos aspectos que os autores destacam como trabalho futuro é a possibilidade de que usuários externos possam controlar remotamente os dispositivos heterogêneos como resposta a notificação de um evento.

---

<sup>3</sup> *Fog Computing*, ou Computação em Névoa, define a arquitetura que estende a capacidade computacional e o armazenamento da nuvem para as camadas de acesso da rede, permitindo que os dados sejam analisados e transformados em informações ou em ações antes de serem simplesmente transmitidos.<sup>13</sup>

<sup>4</sup> *System-on-a-chip*, ou simplesmente *SoC*, é um termo utilizado para caracterizar um circuito integrado que engloba todos os componentes de um computador convencional, tais como CPU, GPU (processador gráfico), memória RAM, módulo Wi-Fi, interfaces externas etc. O Raspberry Pi é um dos mais populares exemplares de *SoC*

<sup>5</sup> RaspberryPi - <<https://www.raspberrypi.org/>>

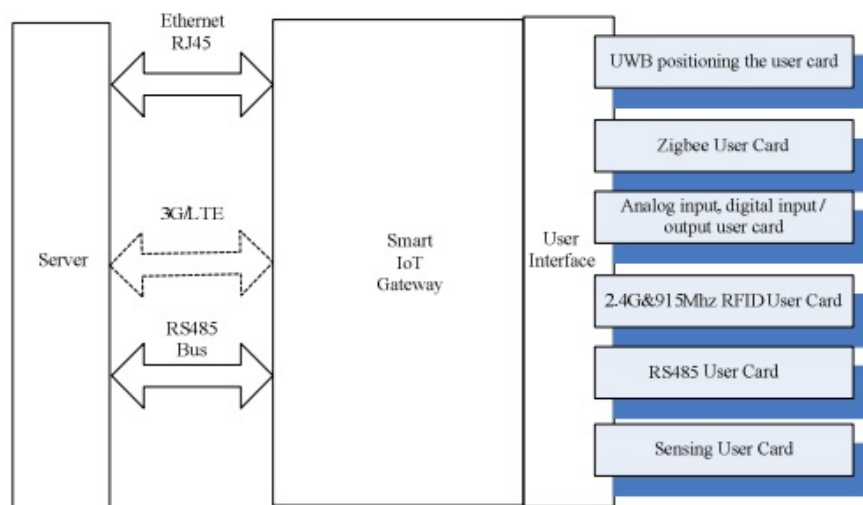


Figura 3 – Arquitetura do Smart IoT Gateway. <sup>15</sup>

## 4 Solução desenvolvida

Esta seção detalha os objetivos e decisões arquiteturais. Foi desenvolvida uma solução de software de *Smart Gateway IoT* que está disponibilizada no Github, capaz de receber dados através de uma rede utilizando o protocolo MQTT (4.1.1) de um dispositivo previamente cadastro e armazenar essas informações. Os dados recebidos são analisados para a estrutura que irá definir a execução ou não de um fluxo de notificação através de SMS (4.1.1) para um número definido.

### 4.1 Tecnologias Utilizadas

- Node.js <sup>6</sup> v6.x;
- TypeScript <sup>7</sup> v2.3 com transpile para ES6;
- TSLint <sup>8</sup> v4.x com recomendações gerais padrão;
- Jest <sup>9</sup> para teste unitário e cobertura;
- AngularJS <sup>10</sup> v1.6 para o front end da aplicação ;
- MongoDB <sup>11</sup> para persistência;
- MQTT (4.1.1) como protocolo de comunicação entre os sensores e o Gateway;
- SMS (4.1.1) como serviço de mensageria;

<sup>6</sup> NodeJS - <<https://nodejs.org/en/>>

<sup>7</sup> Typescript - <<http://www.typescriptlang.org/>>

<sup>8</sup> TSLint - <<https://palantir.github.io/tslint/>>

<sup>9</sup> Jest - <<https://facebook.github.io/jest/>>

<sup>10</sup> AngularJS - <<https://angularjs.org/>>

<sup>11</sup> MongoDB - <<https://www.mongodb.com/>>

- ExpressJS <sup>12</sup> como framework web Node.js

O Node.js foi escolhido por conta de seu baixo consumo de memória e processamento, além da sua característica de Non-Blocking IO <sup>17</sup>, garantindo que possamos servir mais clientes com menos recursos, objetivo essencial para aplicações que podem ser executadas em um Raspberry Pi por exemplo.

A escolha pelo TypeScript <sup>13</sup>, linguagem que é um superset do Javascript padrão foi motivada por garantir uma estrutura tipada, de forma que a manutenção do código fosse facilitada e as regras de negócio pudessem estar ligadas a um contrato de objeto.

Já o AngularJS <sup>14</sup> foi escolhido por ser uma tecnologia que funciona com javascript nativo, sem necessidade de nenhum pós-processador para servir a aplicação aos clientes, permitindo o seu uso diretamente entre os arquivos estáticos do mesmo servidor Node.js que expõe a aplicação. Além disso, contou como um ponto para a escolha, a experiência prévia da equipe no desenvolvimento com esta tecnologia.

#### 4.1.1 Protocolos de comunicação

SMS, abreviação de *Short Message Service*, é um serviço de troca de mensagens curtas de textos que permite o envio de mensagens para aparelhos celulares, conforme os padrões definidos no GSM (*Global System for Mobile Communications*). Seu uso é bastante popular (cerca de 3,6 bilhões de usuários) <sup>18</sup> e ubíquo (presente em praticamente qualquer país) <sup>19</sup>. Além do que seu uso é fácil e barato, uma vez que o usuário receptor não requer conexão com a Internet para receber a mensagem, bastando possuir sinal com a rede de telefonia.

O MQTT, abreviação de *Message Queue Telemetry Transport*, é um protocolo de comunicação altamente voltado para IoT. Ele foi arquitetado para ser um sistema de mensageria leve do tipo publisher/subscriber, para rodar em dispositivos limitados, tanto do ponto de vista da quantidade de memória para execução do programa, quanto do ponto de vista da conectividade. Redes lentas ou com alta latência não são problemas para esse protocolo. Geralmente, sensores IoT podem residir em locais extremamente hostis do ponto de vista de infra-estrutura de conexão de dados.

## 4.2 Modelo de Dados

O modelo de dados foi concebido com a intenção de tornar as etapas do processo altamente plugáveis e customizáveis no curto e longo prazo, garantindo as funcionalidades básicas do MVP <sup>15</sup> executado e a possibilidade de extensibilidade no futuro com retrocompatibilidade.

Nesta modelagem, a entidade **Dispositivo** identifica um aparelho qualquer que se conecte ao gateway para transmitir informações, o aparelho deve definir um ID de conexão que é cadastrado nesta entidade. Com relação direta a entidade **Dispositivo**, e a entidade **Trigger** modela um gatilho clássico, composto de condição e ação. Nesta perspectiva, a condição é a entidade **Operação** que permite avaliar os valores recebidos e identificar se eles

<sup>12</sup> ExpressJS - <<http://expressjs.com/>>

<sup>13</sup> Typescript - <<http://www.typescriptlang.org/>>

<sup>14</sup> AngularJS - <<https://angularjs.org/>>

<sup>15</sup> *Minimum Viable Product*, ou Produto Mínimo Viável, é um termo utilizado para caracterizar um produto com as features necessárias para os primeiros clientes <sup>20</sup>



são iguais ou estão no intervalo de valor definidos por esta entidade. A entidade **Evento** por sua vez é a ação, onde é definido o que acontecerá caso a condição seja atendida, por exemplo, o envio de SMS para um destinatário com o determinado texto, ver Figura 4.

Além das entidades que servem diretamente a execução do fluxo principal da aplicação, temos 3 entidades de funções secundárias. **Configuracao** é responsável por armazenar os dados utilizados para configurar a aplicação, como a configuração das credenciais para uso do serviço de SMS. As entidades **HistoricoEvento** e **HistoricoMensagem** são responsáveis por armazenar os dados recebidos e os eventos executados.

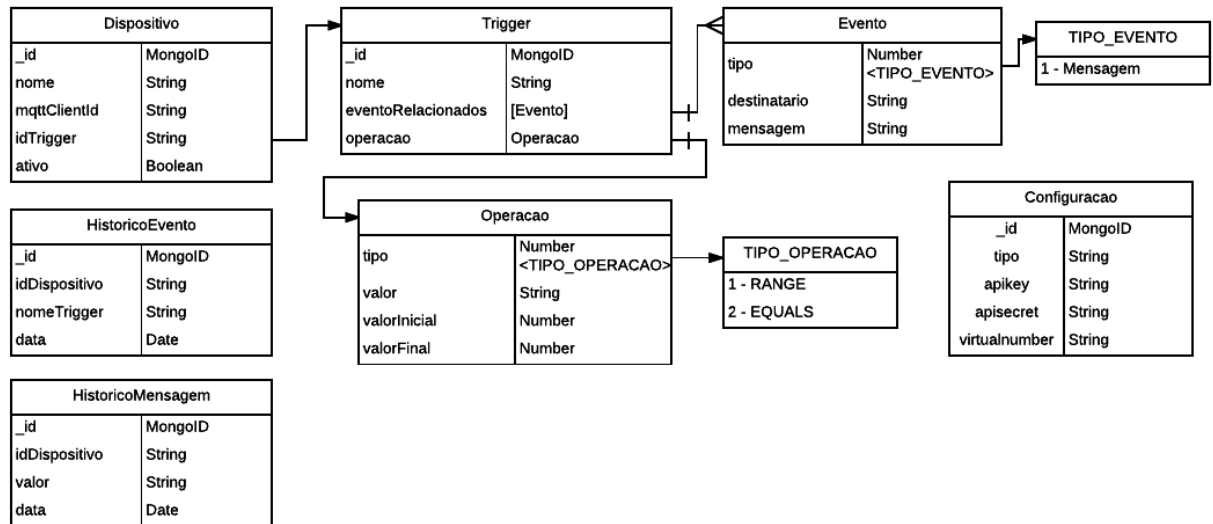


Figura 4 – Representação esquemática da modelagem de dados.

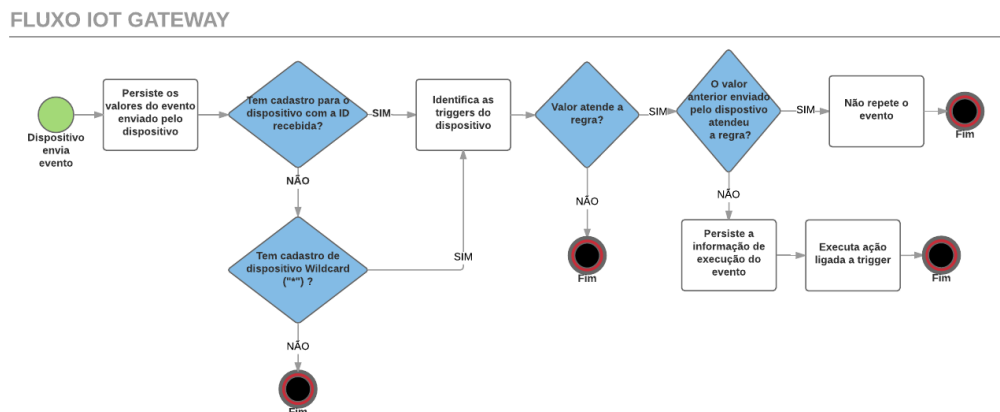


Figura 5 – Representação esquemática do fluxo Gateway IoT.

A modelagem desenvolve uma estrutura sequencial que parte da identificação do dispositivo, análise dos gatilhos ligados a este dispositivo, avaliação da operação lógica e liberação para execução do evento, ver Figura 5.



### 4.3 Requisitos funcionais

Os requisitos funcionais levados em consideração nesse trabalho foram:

- Cadastro de dispositivos
- Conexão de dispositivos via protocolo MQTT
- Visualização e persistência de eventos enviados pelos dispositivos
- Envio de mensagens via SMS
- Configuração de número(s) de celular como destinatário(s) de SMS enviada
- Cadastro de triggers

### 4.4 Requisitos não funcionais

Requisitos não funcionais, como os de extensibilidade e de retrocompatibilidade, agregam muito esforço no desenho de um MVP, mas são aspectos que em hipótese alguma podem ser desconsiderados.

Uma vez que a linguagem escolhida (o Typescript) permite o uso conceitos de Orientação à Objetos (tais como abstrações, tipagens, interfaces e herança), a extensibilidade pode ser garantida por meio da inclusão de novos módulos Typescript e pela extensão do modelo de entidades. Ambos os módulos já existentes e os novos coexistiriam sem muitas dificuldades, garantindo assim a retrocompatibilidade.

O componente `EventoApi.ts` <sup>16</sup> é responsável pelo chaveamento entre as implementações de envio de evento. Vejamos abaixo um trecho de código desse componente:

---

#### Listing 1 EventoApi.ts

---

```
1 import IEvento from '../model/IEvento';
2 import NexmoSMSApi from '../externalapi/NexmoSMSApi';
3 import TIPO_EVENTO from '../model/TIPO_EVENTO';
4 import Configuracao from '../model/Configuracao';

6 /**
7  * Api responsavel por tratamento e execucao de eventos
8  */
9 class EventoApi {

11     private smsApi: NexmoSMSApi;

13     constructor() {
14         this.smsApi = new NexmoSMSApi();
15     }

17     /**
18      * Executa a acao relacionada ao evento
19      *
20      * @param evento a ser executado
21      */
```

---

<sup>16</sup> <https://github.com/RicardoRFaria/IoT-Gateway/blob/master/src/api/EventoApi.ts>

```

22     public executarEvento(evento: IEvento) {
23         if (evento.tipo === TIPO_EVENTO.MENSAGEM) {
24             this.enviarSMS(evento);
25         }
26     }

28     private enviarSMS(evento: IEvento) {
29         // Implementacao particular do envio de SMS que faz uso do
           atributo 'smsApi'...
30     }
31 }

33 export default EventoApi;

```

---

Em relação ao componente acima, caso seja necessário adicionar uma nova forma de envio de evento, basta:

- Injetar a dependência (`import`) (linhas 1 a 4);
- Incluir um novo atributo (linha 11);
- Instanciar o objeto no construtor (linha 14);
- Alterar o método `executarEvento`, fazendo o devido chaveamento conforme o tipo da implementação (linha 22);
- Criar uma `function` específica para tratar o novo tipo de envio.

As configurações específicas da nova forma de envio de eventos poderiam facilmente ser acomodadas na entidade existente de configuração, `Configuracao`, uma vez que a natureza *schemaless* do MongoDB permite esse tipo de cenário.

A compatibilidade com dispositivos cujo hardware possui poucos recursos computacionais, tais como o Raspberry Pi, é o principal requisito não funcional.

## 5 Resultados

Como resultados deste trabalho, podemos destacar o fomento à discussão de soluções que tornem o desenvolvimento para IoT mais simples, removendo a necessidade de codificação de softwares para receber os eventos para cada projeto, conseguindo resultados em um menor tempo. E como principal ponto o desenvolvimento do software que atenda requisitos de um gateway, performático e portátil, abaixo demonstraremos algumas imagens do sistema funcionando.

Na Figura 6 pode-se observar a aplicação sendo inicializada em um Raspberry Pi 3 Model B v1.2 (ver Figura 7), onde a porta TCP 3000 escutará as requisições Web e a porta TCP 1885 onde os dispositivos se conectarão pelo protocolo MQTT.

Na Figura 8 temos a tela inicial da aplicação com algumas informações a respeito da aplicação.

Na Figura 9 vemos o cadastro de uma *trigger* com operação lógica de intervalo, onde o evento será executado apenas caso o valor enviado pelo dispositivo esteja entre 1

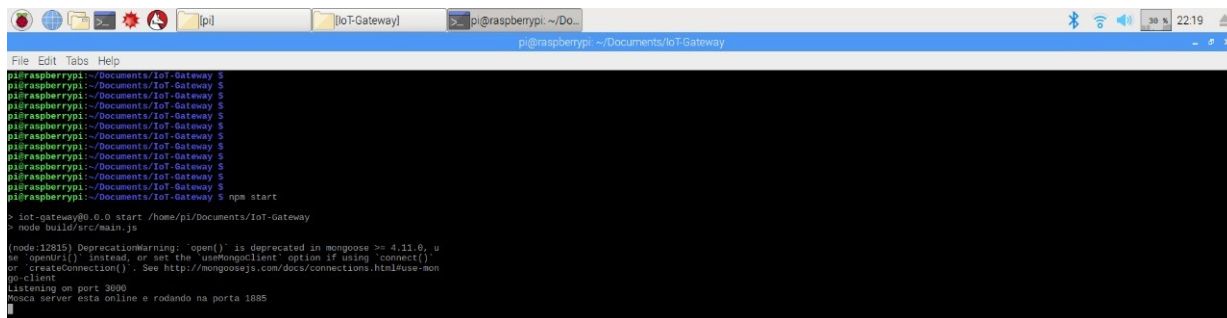


Figura 6 – Representação execução aplicação.

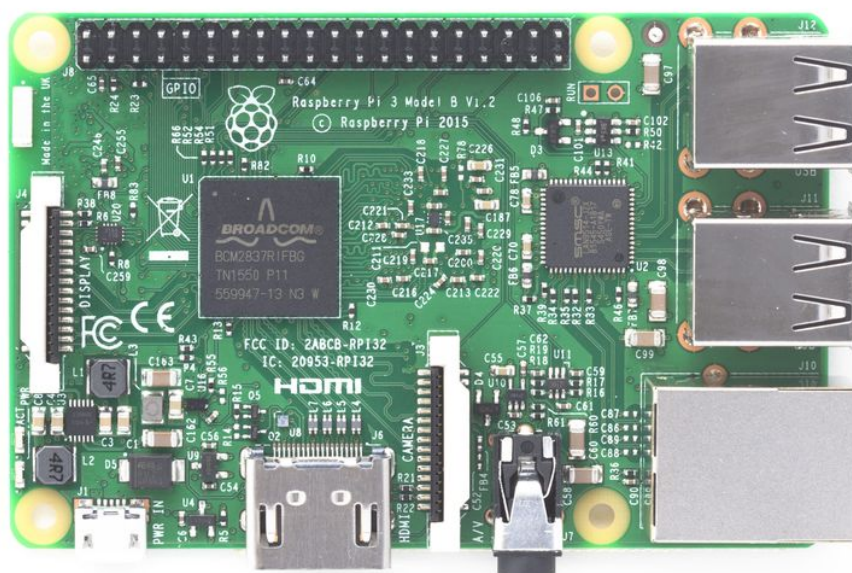


Figura 7 – Exemplar do *Raspberry Pi 3 Model B v1.2*, o SoC utilizado nesse trabalho

e 10. Um cenário claro de uso desta funcionalidade seria o uso do gateway para enviar notificações caso um sensor leia informações de um cenário crítico.

Na Figura 10 temos a listagem de dispositivos cadastrados, onde o dispositivo de nome "Celular com MyMqtt" está cadastrado, mas sua identificar é "\*" que permite que qualquer dispositivo seja identificado por este cadastrado.

Na Figura 11 temos um dispositivo android conectado ao gateway utilizando MQTT e enviando eventos para análise.

Na Figura 12 visualizamos o log da aplicação de 2 eventos, um em que o cenário



Figura 8 – Representação visual da tela dos dados básicos.

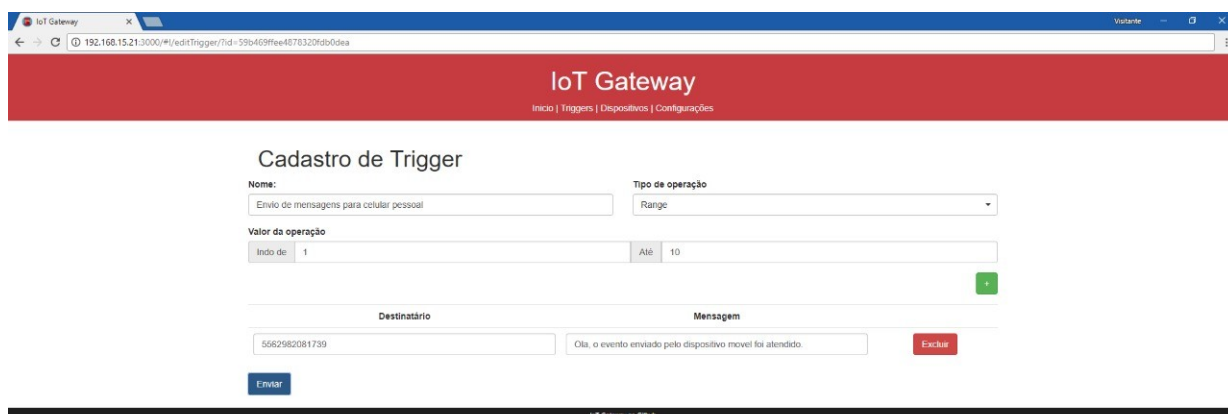


Figura 9 – Representação visual da tela de cadastro de trigger.



Figura 10 – Representação visual da tela de cadastro de dispositivos.

não foi atendido, e outro em seguida onde o mesmo foi atendido e enviou o SMS.

Na Figura 13 o SMS recebido pelo celular após o evento atendido.

Todos os eventos atendidos e dados recebidos são armazenados no banco de dados da aplicação para consulta posterior.

## 6 Considerações finais

Durante o desenvolvimento desse trabalho, podemos concluir que a possibilidade de soluções usando IoT é extremamente vasta e extensa. A indústria, como um todo, está aquecida e pretende absorver toda gama de demanda de eventuais serviços que envolvam esse tipo de tecnologia.

Os Gateways IoT são um ramo da Computação que é relativamente novo, algumas definições não possuem uma consistência desejável e costumam divergir muito de autor para autor. O que caracteriza um certo aspecto positivo, tal que ainda há muito espaço para que mais trabalhos possam ser feitos e consolidados na indústria de IoT.

Existem tecnologias Web modernas, principalmente as baseadas em Javascript, que propiciam a construção de softwares voltado para IoT de forma fácil, produtiva, testável e manutenível.

Na elaboração desse projeto, no que tange a perspectiva da escrita do software, o maior desafio foi dos pontos de vista da elaboração arquitetural e de modelagem de dados, de forma que após as definições, a escrita do código foi relativamente simples.

## 6.1 Trabalhos futuros

Como trabalhos futuros, destacamos um suporte a mais protocolos de comunicação com dispositivos e outras funções existentes em outros gateways, como a transmissão posterior das informações armazenadas para a cloud e suporte a novos eventos como, envio de email, ligações e até envio de informações para outros dispositivos, assim tornando o gateway como um dispositivo não só passivo, mas como atuante na rede em que eles está inserido.

Outro aspecto que poderia ser melhorado é o suporte a outros operadores lógicos, tais como, maior que ( $>$ ), maior ou igual que ( $\geq$ ), menor que ( $<$ ), menor ou igual que ( $\leq$ ), diferente de ( $\neq$ ) etc.

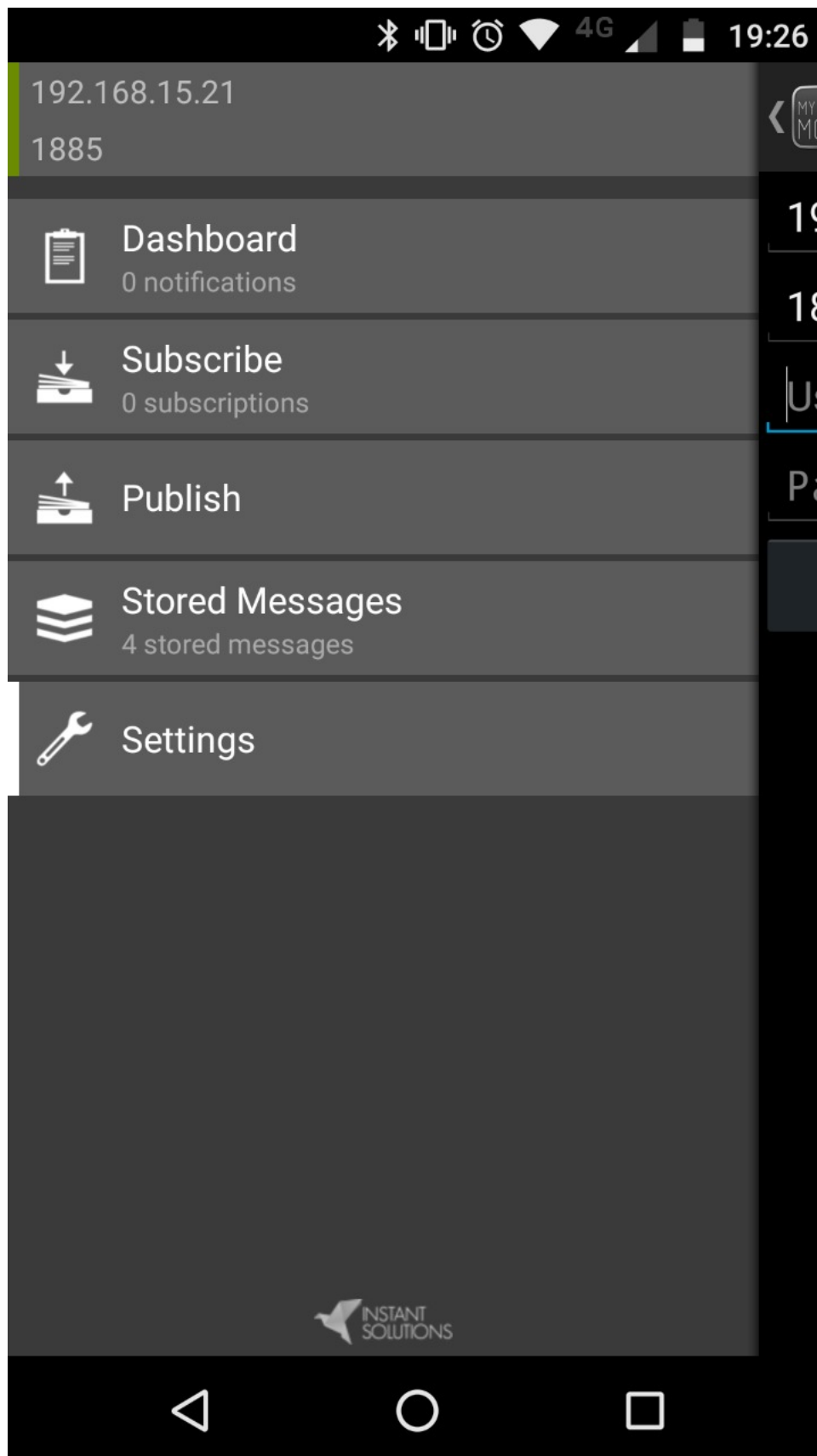
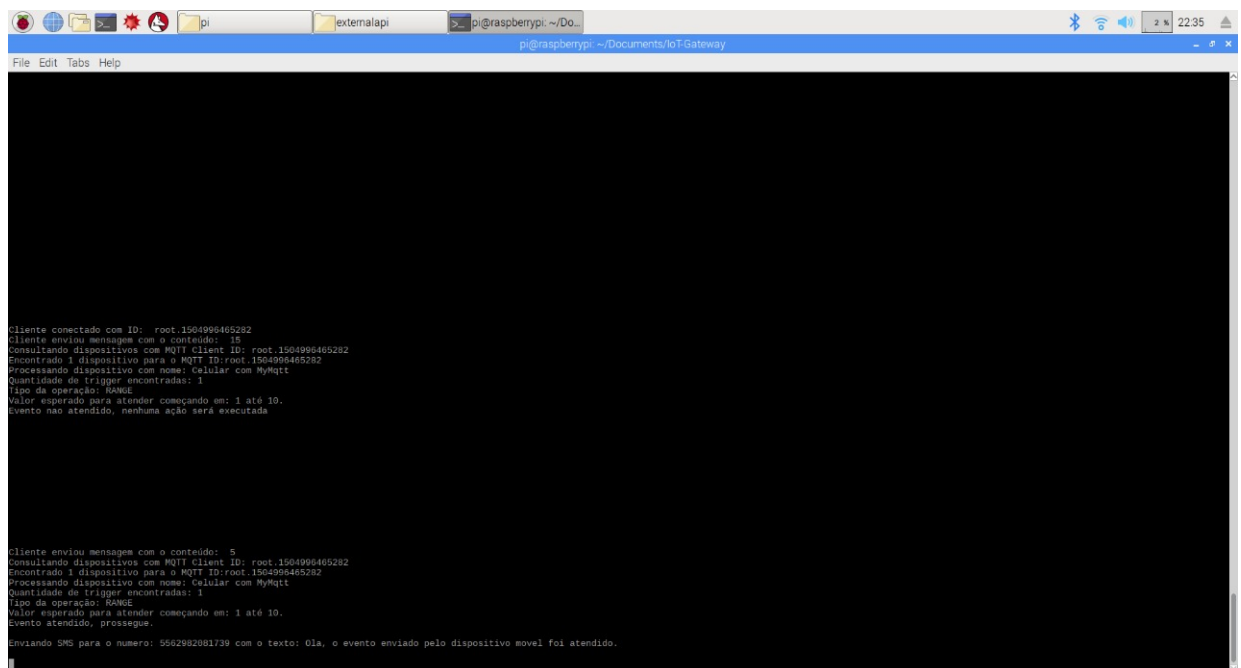


Figura 11 – Representação visual do dispositivo Android utilizando protocolo MQTT.



The image shows a terminal window on a Raspberry Pi. The window title is 'pi@raspberrypi: ~/Documents/loT-Gateway'. The terminal output displays two MQTT events. The first event shows a client with ID 'root.1504996465282' sending a message with content '15'. The system then searches for devices, finds one with the same ID, and processes it as a 'Celular com MyMqtt'. It identifies a 'RAWID' trigger and checks if the expected value (1) is within the range (1 to 10). Since the event is not attended, no action is executed. The second event shows a client sending a message with content '5'. The system finds the same device, processes it as a 'RAWID' trigger, and checks the expected value (5) against the range (1 to 10). Since the event is attended, it proceeds to send an SMS to the number '5502902081739' with the text 'Ola, o evento enviado pelo dispositivo movel foi atendido.'

```
pi@raspberrypi: ~/Documents/loT-Gateway

Cliente conectado com ID: root.1504996465282
Cliente enviou mensagem com o conteúdo: 15
Consultando dispositivos com MQTT Client ID: root.1504996465282
Encontrado 1 dispositivo para o MQTT ID:root.1504996465282
Processando dispositivo com nome: Celular com MyMqtt
Quantidade de trigger encontradas: 1
Tipo da operação: RAWID
Valor esperado para atender começando em: 1 até 10.
Evento nao atendido, nenhuma ação será executada

Cliente enviou mensagem com o conteúdo: 5
Consultando dispositivos com MQTT Client ID: root.1504996465282
Encontrado 1 dispositivo para o MQTT ID:root.1504996465282
Processando dispositivo com nome: Celular com MyMqtt
Quantidade de trigger encontradas: 1
Tipo da operação: RAWID
Valor esperado para atender começando em: 1 até 10.
Evento atendido, prossigue.
Enviando SMS para o numero: 5502902081739 com o texto: Ola, o evento enviado pelo dispositivo movel foi atendido.
```

Figura 12 – Representação visual do console ao receber eventos.



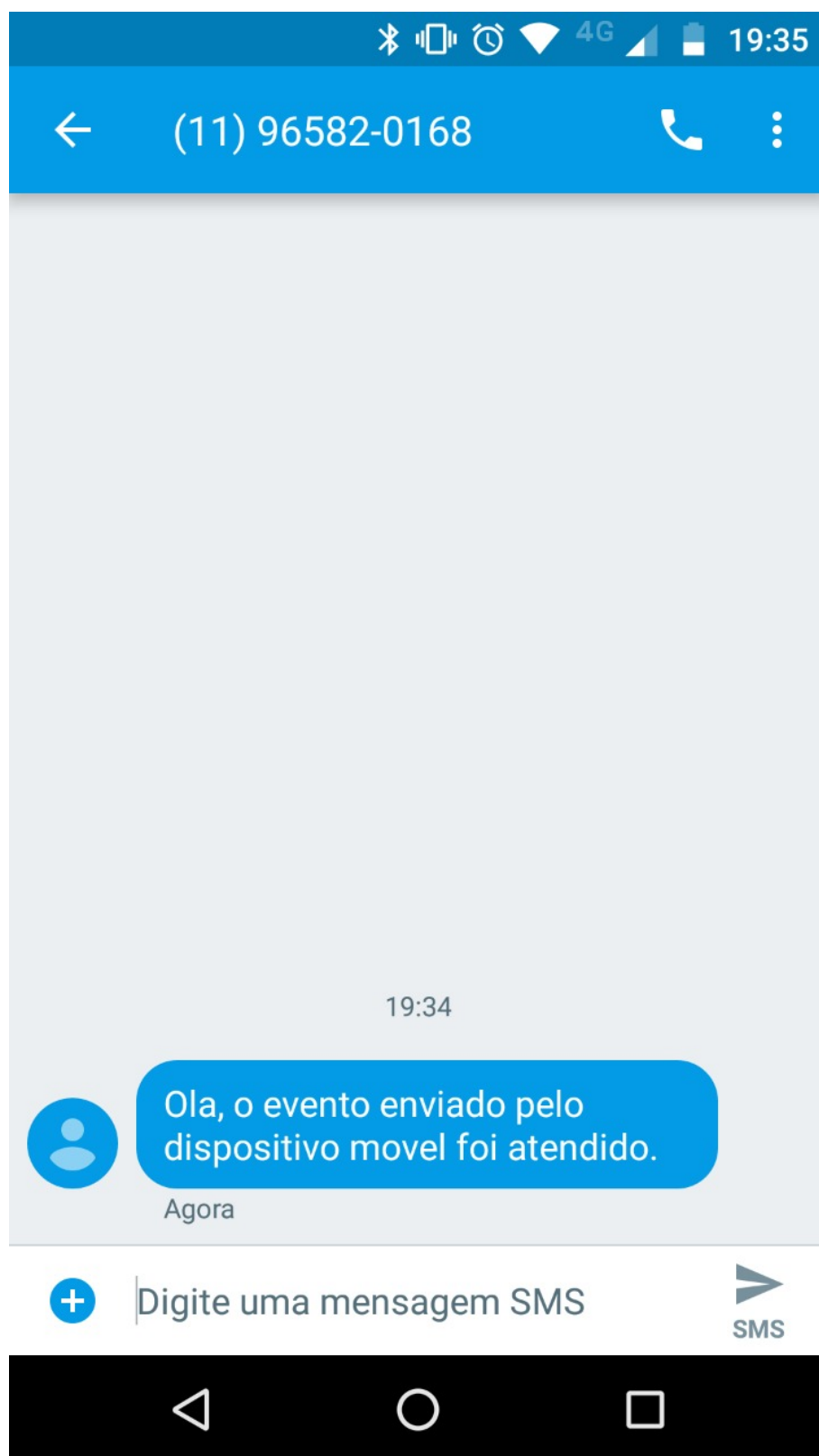


Figura 13 – Representação visual do SMS recebido.

## Referências

- 1 ANDRIOLLI, A. *Gateways serão fundamentais para segurança em projetos de IoT*. [S.l.]: ComputerWorld. [Http://computerworld.com.br/gateways-serao-fundamentais-para-seguranca-em-projetos-de-iot](http://computerworld.com.br/gateways-serao-fundamentais-para-seguranca-em-projetos-de-iot). Citado 2 vezes nas páginas 2 e 3.
- 2 BCG Perspectives, The Boston Consulting Group. *Winning in IoT: It's All About the Business Processes*. Citado na página 2.
- 3 Forbes. *Internet Of Things Market To Reach \$267B By 2020*. [Https://www.forbes.com/sites/louiscolumnbus/2017/01/29/internet-of-things-market-to-reach-267b-by-2020](https://www.forbes.com/sites/louiscolumnbus/2017/01/29/internet-of-things-market-to-reach-267b-by-2020). Citado na página 2.
- 4 Warren Kuris. *Designing customised intelligent IoT gateway*. [S.l.]: EET Asia. [http://archive.eetasia.com/www.eetasia.com/ART\\_8800719530\\_590626\\_TA\\_8734b523.HTM](http://archive.eetasia.com/www.eetasia.com/ART_8800719530_590626_TA_8734b523.HTM). Citado na página 2.
- 5 Brasil Escola. *INTERNET*. [S.l.]: Monografias Brasil Escola. Citado na página 3.
- 6 ASHTON, K. That "Internet of Things" Thing. *RFiD Journal*, v. 22, p. 97–114, 2009. Citado na página 3.
- 7 Amazon Web Services. *Internet das Coisas*. [S.l.]: Amazon Web Services, Inc. Citado na página 3.
- 8 Ahmed Banafa. *Three Major Challenges Facing IoT*. [S.l.]: IEEE. Citado na página 3.
- 9 David Roe. *7 Big Problems with the Internet of Things*. [S.l.]: CMS WiRE. Citado na página 3.
- 10 CHEN, H.; JIA, X.; LI, H. A brief introduction to IoT gateway. In: *Communication Technology and Application (ICCTA 2011), IET International Conference on*. [S.l.: s.n.], 2011. Citado na página 3.
- 11 ROUSE, M. *WhatIs.com: IoT Gateway*,. 2017. Citado na página 4.
- 12 KONSEK, H. *The Architecture of IoT Gateways*. [S.l.]: DZone, 2015. [Https://dzone.com/articles/iot-gateways-and-architecture](https://dzone.com/articles/iot-gateways-and-architecture). Citado na página 4.
- 13 CIO. *Fog Computing é o novo paradigma para a Internet das Coisas, diz Cisco*. [S.l.]: CIO From IDG. Citado na página 5.
- 14 RS Online. *UK RS Online - Raspberry Pi 3 Model B SBC*. [Http://uk.rs-online.com/web/p/processor-microcontroller-development-kits/8968660/](http://uk.rs-online.com/web/p/processor-microcontroller-development-kits/8968660/). Citado na página 5.
- 15 Shang Guoqiang, Chen Yanming, Zuo Chao. *Design and Implementation of a Smart IoT Gateway*. [S.l.]: IEEE.org. Citado 2 vezes nas páginas 5 e 6.
- 16 Diana Cecilia Yacchirema Vargas and Carlos Enrique Palau Salvador. *Smart IoT Gateway For Heterogeneous Devices Interoperability*. [S.l.]: IEEE.org, 3900-3906 p. <http://ieeexplore.ieee.org/document/7786378/>. Citado na página 5.

- 17 Node.js Foundation. *Node.js - Overview of Blocking vs Non-Blocking*. <https://nodejs.org/en/docs/guides/blocking-vs-non-blocking/>. Citado na página 7.
- 18 Tomi T Ahonen. *Time to Confirm some Mobile User Numbers: SMS, MMS, Mobile Internet, M-News*. 2011. <http://communities-dominate.blogs.com/brands/2011/01/time-to-confirm-some-mobile-user-numbers-sms-mms-mobile-internet-m-news.html>. Citado na página 7.
- 19 International Telecommunications Union. [S.l.]: International Telecommunications Union, 2015. <https://www.itu.int/en/ITU-D/Statistics/Documents/facts/ICTFactsFigures2015.pdf>. Citado na página 7.
- 20 Eric Ries. *Venture Hacks interview: "What is the minimum viable product?"*. 2009. <http://www.startuplessonslearned.com/2009/03/minimum-viable-product.html>. Citado na página 7.