UNIVERSIDE DE SÃO PAULO
ESCOLA DE ENGENHARIA DE SÃO CARLOS
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

Smart Meter – Uma abordagem voltada para IoT

Aluna: Heloisa Junqueira Barbosa

Orientador: Prof. Dr. Ivan Nunes da Silva

São Carlos

2016

HELOISA JUNQUEIRA BARBOSA

Smart Meter – Uma abordagem voltada para IoT

Trabalho de conclusão de curso apresentado à Escola de Engenharia de São Carlos, da Universidade de São Paulo
Curso de Engenharia Elétrica com ênfase em Sistemas de Energia e Automação

Orientador: Prof. Dr. Ivan Nunes da Silva

São Carlos
2016

Barbosa, Heloisa Junqueira
B482s Smart Meter - Uma abordagem voltada para IoT / Heloisa Junqueira Barbosa; orientador Ivan Nunes da Silva. São Carlos, 2016.

Monografia (Graduação em Engenharia Elétrica com ênfase em Sistemas de Energia e Automação) -- Escola de Engenharia de São Carlos da Universidade de São Paulo, 2016.

1. Smart Meter. 2. IoT. 3. Sistemas embarcados. 4. Serviços em nuvem . I. Título.

FOLHA DE APROVAÇÃO

Nome: Heloisa Junqueira Barbosa

Título: "Smart Meter - Uma abordagem voltada para IoT"

Trabalho de Conclusão de Curso defendido e aprovado em 22 / 11 / 2016,

com NOTA 10,0 (DEZ, ZERO), **pela Comissão Julgadora:**

Prof. Associado Ivan Nunes da Silva - Orientador - SEL/EESC/USP

Prof. Associado Evandro Luis Linhari Rodrigues - SEL/EESC/USP

Mestre Rafael Guedes Lang - Doutorando - SEL/EESC/USP

Coordenador da CoC-Engenharia Elétrica - EESC/USP:
Prof. Associado José Carlos de Melo Vieira Júnior

Agradecimentos

Ao meu orientador, Prof. Dr. Ivan Nunes da Silva, pelo apoio e suporte dado durante o desenvolvimento do projeto e pelo conhecimento transmitido através da graduação.

À minha família e a Valda Rastelli por todo o suporte e ajuda durante minha trajetória.

À *Toradex*, principalmente ao Guilherme, pelo suporte dado durante o período de estágio, buscando sempre ajudar e a extrair sempre melhor da equipe.

Ao *Warthog Robotics*, por tudo o que representou durante a minha graduação, bem como aos amigos que vieram a somar em meu desenvolvimento, em especial ao Rafael Lang.

Resumo

O cenário energético brasileiro enfrenta atualmente problemas tais como o crescimento do consumo, preocupação crescente com a sustentabilidade e limitações na capacidade de expansão da infraestrutura de geração de energia elétrica, fatores estes que incentivam o uso de tecnologias que venham a contribuir para a economia de energia. Visando munir o consumidor de uma ferramenta para análise precisa e instantânea, para monitoramento do consumo de energia elétrica, foi desenvolvido o projeto de um *Smart Meter*. Visando o cenário de *IoT*, este projeto foi concebido por meio da utilização de um módulo embarcado, que realizou o monitoramento do consumo de energia de uma casa. Estes dados de consumo foram enviados para um serviço em nuvem, processados e posteriormente exibidos através de gráficos de fácil compreensão e interpretação para o consumidor final, trazendo para realidade a proposta de conectar dispositivos na nuvem e obter ferramentas para uma análise de dados mais otimizada. Neste projeto tais ferramentas possibilitaram uma tomada de medidas de economia de energia mais precisas e objetivas ao longo do mês. A utilização módulos embarcados e serviços em nuvem possibilitaram a criação de uma solução que utiliza ferramentas disponíveis no mercado, utilizadas em escala industrial e que possibilitam uma redução no tempo de *Time-to-Market* (TtM) de um produto.

Palavras chave: Smart Meter, IoT, Sistemas embarcados, Serviços em nuvem

Abstract

The current Brazilian energy scenario is facing, nowadays, problems such as: the growth of consumption, problems concerning sustainability, and limitations in the scalability of the infrastructure of power generation, factors that encourage the use of technologies that will contribute to energy savings. Aiming to give the consumer a tool capable of precise analysis and real time, the design of a Smart Meter was developed. Focused on the IoT scenario, this project was conceived through the use of an embedded module that will carry out the monitoring of energy consumption of a house. These consumption data will be sent to a cloud service, processed and then displayed through charts that will be easily understood and interpreted by the final consumer, bringing to reality the proposal to connect devices in the cloud and get tools for a more optimized data analysis. With this Project, such tools will enable more accurate and objective actions for energy saving, throughout the month. The use of embedded modules and cloud services enabled the creation of a solution that uses tools available in the Market and used on an industrial scale and which enables a reduction in the Time-to-Market (TtM) of a product.

Keywords: Smart Meter, IoT, Embedded systems, Cloud services.

Lista de Figuras

Lista de Tabelas

Lista de Algoritmos

Lista de abreviações

DARPA - *Defense Advanced Research Projects Agency*

IoT - *Internet of Things*

M2M - *Machine-to-Machine*

MIT - *Massachusetts Institute of Technology*

TtM- *Time-to-Market*

SoM - *System-on-Module*

NIST - *The National Institute of Standards and Technology*

HTTP - *Hypertext Transfer Protocol*

SoC - *System On Chip*

SoM – *System on Module*

GPIO - *General Purpose Input/Output*

I2C - *Inter-Integrated Circuit*

SBC - *Single Board Computer*

IDE - *Integrated Development Environment*

Sumário

1- Introdução

Neste cenário de crescimento da população, muito maior que a média de 2%, a necessidade de mais e mais energia é cada vez mais crescente[1]. Em um estudo realizado pela *World Energy Council* (WEC, em português Concelho Mundial de Energia) aponta que, se nenhuma alteração dos modos de consumo for realizada, a demanada mundial de energia em 2020 seria de 50-80% superior se comparada a 1990[2], dados referentes ao ano de 1995.

Esta necessidade gera um cenário de dependência cada vez maior, um exemplo disto são os efeitos que o corte de energia de 24 horas poderia causar em uma cidade tais como, o corte total de elevadores e computadores[1], interrupção da linha de produção das indústrias, resultando em um grande prejuízo financeiro.

Em muitos países ocidentais esta preocupação com problemas ambientais e energéticos [3] está se concretizando através de ações do próprio governo, um exemplo é a Inglaterra. Onde em dezembro de 2009 o *UK Department for Energy and Climate Change* (DECC, em português Departamento de Energia e Alterações Climáticas do Reino Unido) anunciou a intenção de lançar, no Reino Unido até 2020, “Smart Meters” (em português, medidores inteligentes) que fossem acompanhados de displays e exibissem o consumo instantâneo de energia. Decisão esta justificada pela afirmação de que estes medidores irão fornecer dados instantâneos de consumo, os quais irão ajudar o consumidor a controlar seu consumo, economizar na conta de luz e reduzir emissões de poluentes[5].

Visando tal cenário, este projeto consistiu em uma proposta de um Smart Meter, que utilizando sistemas embarcados, realiza o monitoramento do consumo de energia de uma casa.

1.1- Objetivos

Esta proposta de conectar objetos, neste caso sensores, a fim de oferecer ferramentas para tomada de decisões, está atualmente sendo cada vez mais explorada pelas empresas por meio do conceito de *Internet of Things* (IoT, em português Internet das Coisas), que consiste no uso da internet para a interconexão de objetos físicos que se comunicam uns com os outros e/ou com os seres humanos, a fim de oferecer um determinado serviço[6]. Apesar do conceito de IoT estar sendo aplicado há algum tempo, como citado por professores do *Massachusetts Institute of Technology* (MIT) há cerca de 20 anos, em cenários tais como monitoramento de casas remotamente, envio de dados de pesquisa do celular para aplicativos de

propaganda, dentre outros. Este cenário ganha força , pois além de propor a conexão de dispositivos à internet, os mesmos sejam utilizados para a extração de diferentes tipos de dados disponíveis[4].

Estes dados servirão para extração de ideias e um consequente aumento da eficiência e melhoria dos serviços e processos por meio, por exemplo, da automatização, robotização, dentre outros[4]. É neste ambiente que se insere tecnologias tais como sistemas embarcados, onde há necessidade de se conectar cada vez mais dispositivos à internet, utilizando um espaço físico e consumo de energia cada vez menores.

Com o intuito de abordar um cenário completo de *IoT*, que vai desde a aquisição dos dados até a disponibilização dos mesmos para análise, foi proposto um projeto que abordasse tal cenário de maneira completa e de fácil reprodução para diferentes necessidades.

Por meio da utilização de sistemas embarcados e computação em nuvem buscou-se apresentar tecnologias que viessem a reduzir o tempo de *Time-to-Market*(TtM) de um produto, o qual consiste no tempo decorrido entre a definição do produto e disponibilidade do produto[7], bem como a utilização de soluções voltadas para aplicações que pudessem ser utilizadas em escala industrial.

1.2- Justificativas

No Brasil, devido a fatores tais como, o crescimento do consumo, preocupação crescente com sustentabilidade e limitações na capacidade de expansão da infraestrutura de geração, o gerenciamento pelo lado da demanda surge como uma alternativa indispensável de planejamento para o setor elétrico[8]. Visando este cenário, o projeto teve como objetivo abordar tecnologias para o monitoramento de energia instantaneamente, bem como a utilização de ferramentas que consigam de forma gráfica e de fácil compreensão exibir estes dados para o consumidor.

Para tal finalidade foram utilizados sensores de corrente que realizam a medição em um conjunto de 4 tomadas, para ilustrar uma possível implementação em um quadro de luz de uma residência. A leitura destes sensores é feita através do módulo Colibri VF50 + placa base Viola Plus da *Toradex*. Estes dados serão posteriormente enviados para a nuvem, processados, tratados e exibidos instantâneamente para o consumidor. Estes dados servirão para análise e identificação dos gastos de uma residência, bem como munir o consumidor de uma ferramenta que o ajude a economizar diariamente energia, de forma objetiva e o mais precisa possível.

Por meio do envio de dados do dispositivo para nuvem, a fim de obter uma ferramenta para análise de dados para uma tomada de decisões mais precisas, procurou-se também abordar o conceito de *IoT* neste projeto. Trazendo para a realidade a tradução deste conceito abordando desde a parte do *Things* em português, coisas), que neste projeto são os sensores de medição de uma casa, até a parte de *Internet* que

irá armazenar e tratar estes dados, disponibilizando-os para uma posterior análise e tomada de decisões. Abaixo está ilustrado o diagrama do projeto que será abordado nos capítulos seguintes.

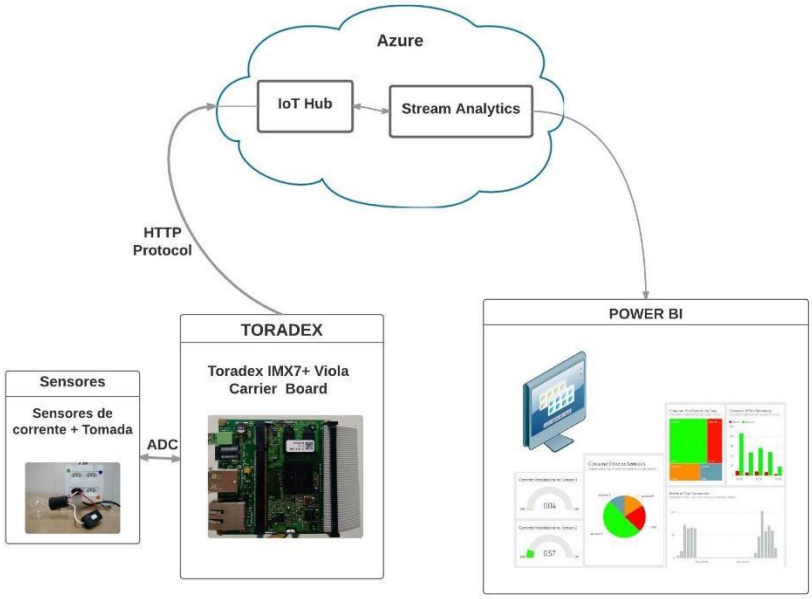


Figura 1: Fluxograma contendo as etapas do projeto Smart Meter[38]

1.3- Organização do trabalho

O trabalho é dividido em cinco capítulos. No primeiro capítulo são apresentados o cenário e uma breve introdução sobre o objetivo deste projeto. No segundo capítulo são abordados os conceitos e fundamentos teóricos utilizados como base de pesquisa para elaboração deste projeto. O terceiro aborda os materiais utilizados bem como os métodos utilizados no desenvolvimento do mesmo. No quarto são apresentados os resultados obtidos bem como a discussão do mesmo. No último capítulo são apresentadas algumas considerações finais e propostas de trabalhos futuros.

2- Embasamento teórico

Nesta seção serão apresentados os temas e linhas de pesquisa e desenvolvimento, os quais foram utilizados como base para o desenvolvimento do projeto.

2.1- Sistemas embarcados

Uma definição para sistema embarcados é difícil e fluida, porém algumas características comuns de tais sistemas são: o fato de serem projetados para executar uma função dedicada [9][10], o que consequentemente resulta em uma limitação de tamanho e processamento tanto de hardware como de software. Resultando em um hardware tipicamente de baixo consumo, memória e funcionalidades reduzidas, e a nível de software possuindo sistemas operacionais mais reduzidos e otimizados para aplicações específicas[9], características estas se comparados a um computador. Porém com o constante avanço da tecnologia algumas destas características evoluíram e tais sistemas estão cada vez mais semelhantes aos computadores de arquitetura x86.

A importância de tais sistemas surge do fato de estarem cada vez mais presentes no dia-a-dia das pessoas, sem que as mesmas se deem conta de que os mesmos sejam caracterizados como sistemas embarcados, como por exemplo micro-ondas, totem de shoppings, sistemas de multimídia em carros, equipamentos médicos, dentre outros [10].

Visando um cenário que necessite de baixo consumo, conectividade e desempenho necessário para uma aplicação dedicada, optou-se pela utilização de sistemas embarcados para aquisição dos dados do sensor de corrente e envio dos mesmos para serviços em nuvem.

2.1.1 Sistema Operacional Embarcado

O Sistema Operacional (SO) utilizado na concepção deste projeto é a distribuição *Ångström* que utiliza o *Kernel Linux*. SO baseados em *Linux* tem como principal característica ser uma plataforma *open-source*, a qual permite que os desenvolvedores contribuam e otimizem tanto o SO e *Kernel*, quanto *drivers* utilizados pelo mesmo, conferindo uma grande quantidade de documentação disponíveis [4].

Por se tratar de SO modular e *open-source*, o mesmo possui ferramentas para customização da imagem, tais como *Yocto Project*, *Open Embedded* [12], dentre outras. Esta construção da própria imagem possibilita a otimização do espaço utilizado pela aplicação a ser desenvolvida, podendo ser geradas imagens que possuem somente as funções básicas que ocupam 4 MB de *SDRAM* e 2 MB de *flash*[13].

Outras razões para o crescimento de imagens baseadas no *Kernel* do *Linux* é o fato de o mesmo suportar uma enorme variedade de dispositivos de hardware, provavelmente mais que qualquer outros SO. Bem como suportar uma enorme variedade de aplicações e protocolos de comunicação e poder ser depurado sem a utilização de programas proprietários. [12]

2.2- Internet das Coisas (IoT)

O conceito de IoT, diferentemente de como é considerado não é um conceito recente, o mesmo começou a ser discutido a 20 anos atrás, onde professores do MIT, descreveram um mundo onde "coisas" (dispositivos ou sensores) são conectados e capazes de compartilhar dados[15], dados estes que serão utilizados para posterior análise e extração ideias.

Contudo, este conceito vem ganhando cada vez mais espaço devido a fatores tais como: aumento da acessibilidade a internet, a um custo cada vez mais baixo, possibilitando cada vez mais a conexão de dispositivos, os quais possuem cada vez mais sensores para aquisição de diferentes dados. O que culmina em uma consequente diminuição dos preços de tais tecnologias, a medida que as mesmas estão sendo cada vez mais difundidas [16].

Neste cenário surge também o conceito de comunicação *Machine-to-Machine* (M2M, em português máquina-máquina), a qual permite estabelecer condições em que o dispositivo troquem informações com uma aplicação, extraindo informações estratégicas via rede de comunicação [15].

Nas figuras abaixo estão exemplificados a ocorrência desta comunicação, onde um dispositivo ou mais, podem enviar diretamente a informação para a nuvem ou onde os mesmos necessitam de um dispositivo intermediário para enviar estes dados, como por exemplo um *Gateway*[15]. Porém em todos estes casos a comunicação é feita autonomamente, ou seja, sem o auxílio ou a interferência humana durante o processo de comunicação.

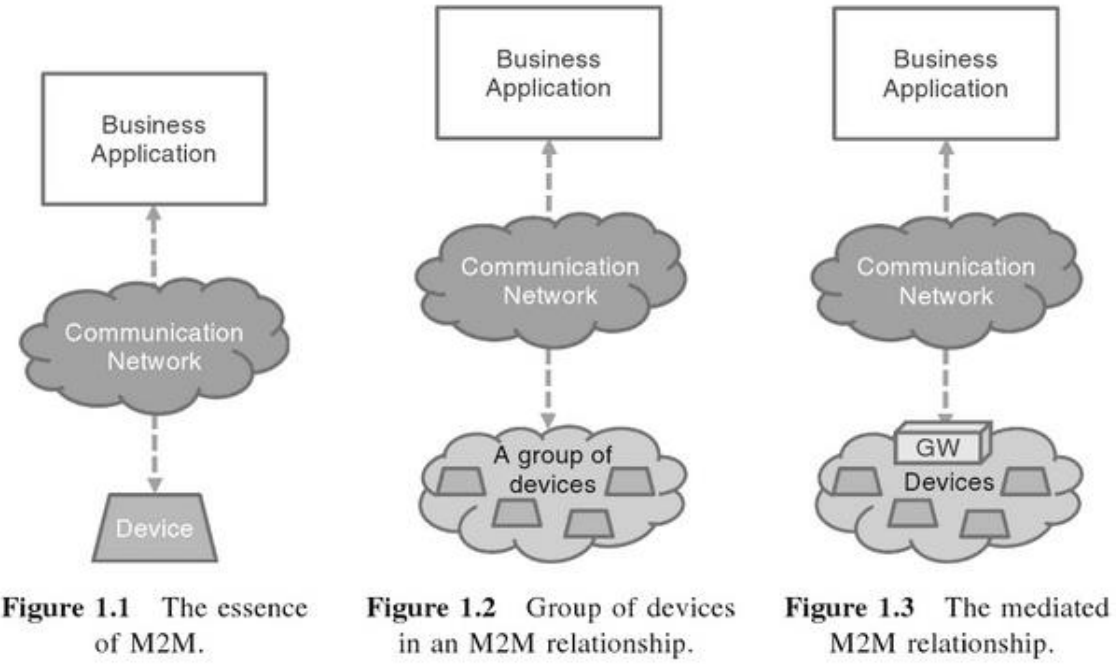


Figure 1.1 The essence of M2M. **Figure 1.2** Group of devices in an M2M relationship. **Figure 1.3** The mediated M2M relationship.

Figura 2: Tipos de comunicação M2M [15]

2.3- Computação em nuvem

Com o rápido desenvolvimento de tecnologias de processamento e armazenamento, os recursos computacionais estão evoluindo cada vez mais e seu custo tornando-se cada vez mais acessíveis. Porém, no meio empresarial o aumento da utilização de tais tecnologias resultam em um aumento proporcional da complexidade do gerenciamento desta infraestrutura, que engloba a arquitetura de informação, dados e software distribuídos, tornando-a cara e necessitando de um alto investimento inicial para sua implementação e custos de posterior manutenção[14]. É neste cenário que surgem serviços para auxiliar e simplificar a utilização de tais recursos, os quais são denominados de plataformas em nuvem.

Segundo *The National Institute of Standards and Technology* (NIST) dos Estados Unidos a definição de computação em nuvem é modelo que possibilita um acesso, convenientemente e sob demanda, a um conjunto compartilhado de serviços computacionais configuráveis (tais como servers, armazenamento, aplicações, serviços, dentre outros) os quais podem rapidamente prover e liberar, com o mínimo esforço de gestão ou serviço, a interação com o servidor[16]. Esta definição traz recursos que são muito atraentes do ponto de vista empresarial, tais como a não necessidade de investimentos iniciais em infraestrutura, alta escalabilidade, fácil acesso, reduzido riscos de negócios e despesas de manutenção, dentre outras vantagens[4].

2.4- Time-to-Market

Levando-se em conta o atual cenário industrial onde os ciclos de vida dos produtos estão se tornando cada vez mais mais curtos, em um mercado cada vez mais exigente e que incentiva cada vez mais o lançameto de novas tecnologias[62]. É neste contexto que torna-se cada vez mais necessária a redução do tempo de *Time-to-Market* (TtM, em português Indicador-chave de desempenho) de um produto, o qual consiste no tempo decorrido entre a definição do produto e disponibilidade deste no mercado consumidor[7].

Tendo em vista a importância desta variável, grandes empresas procuram reunir equipes multifuncionais com a finalidade de reduzir o ciclo de desenvolvimento de novos produtos [61][7]. Estudos realizados na área estimam que uma perda de 50-75% das vendas de computador pessoal em empresas, devido a um traso de 6 a 8 meses no lançamento do produto[62].

2.5 - Sensor de corrente

O sensor de corrente tem como finalidade medir um campo magnético e converter essa medição em tensão, utilizando o princípio do efeito Hall [23]. O princípio físico por de traz do efeito Hall é o efeito da Lei de Lorentz, no qual o sensor consiste em um caminho de condução de cobre, que através da aplicação de corrente, será gerado um campo magnético. Este campo irá experimentar uma força, F, a força de Lorentz, que é normal tanto ao campo quanto a corrente que flui [22], como ilustrado na Figura 2.

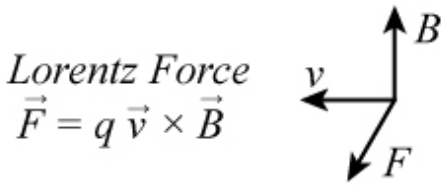


Figura 3: Diagrama ilustrativo da Lei de Lorentz [22]

Em resposta a esta força, os elétrons se movem em um caminho curvo ao longo do condutor, gerando uma tensão que se desenvolve ao longo da placa. Esta tensão de Hall, V_H, obedece à fórmula abaixo e pode ser ilustrado por meio da da Figura 3, o que demonstra que V_H é proporcional à força do campo aplicado, e que a polaridade de V_H é determinada pela direcção do norte ou sul, do campo magnético aplicado[22].

$$V_H = \frac{I B_\perp}{\rho_n q t}$$

V_H é a tensão Hall através da placa,

I é a corrente passando através da placa,

q é a magnitude da carga dos portadores de carga,

ρn é o número de portadores de carga por unidade de volume

t é a espessura da placa

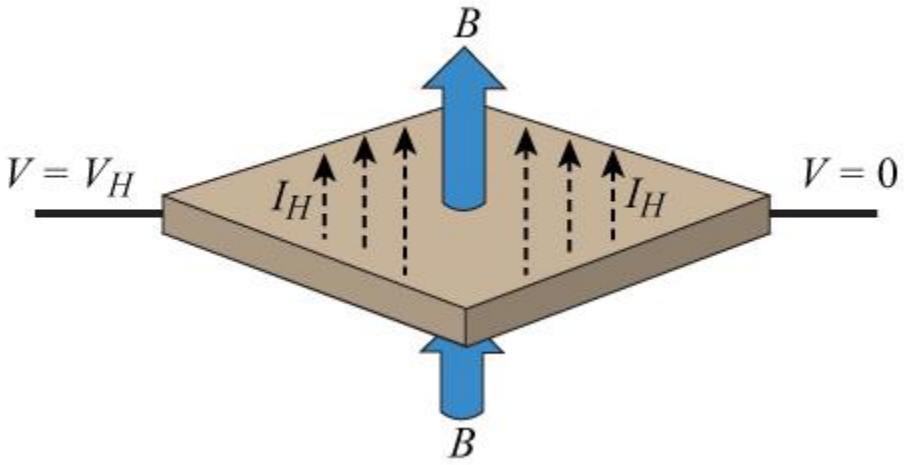


Figura 4: Efeito Hall, onde a seta azul indica o campo magnético que passa perpendicularmente através da placa[22]

Devido aos terminais do caminho condutor serem eletricamente isoladas dos condutores de sinal, permite ao sensor ser utilizado em aplicações que requerem o isolamento eléctrico sem a utilização de opto-isoladores ou outras técnicas de isolamento [23].

3 - Materiais e métodos

A organização deste trabalho, bem como o método de realização do estudo da elaboração deste trabalho serão apresentados e descritos detalhadamente. Posteriormente serão apresentados os resultados obtidos através deste.

3.1 Materiais

Nesta seção serão apresentados os materiais utilizados e abordados no desenvolvimento deste projeto.

3.1.1- Hardware - SoM Toradex

Neste projeto foi utilizado um *System-on-Module* (SoM, em português sistema em módulo) Colibri VF50 da *Toradex* em conjunto com a placa base Colibri Viola Plus também da *Toradex*. Esta solução, considerada de baixo consumo, possui CPU do tipo ARM Cortex™-A5 e 128MB de memória RAM e Flash cada uma [11].

3.1.1.1- Sistema em Módulo VF50

O módulo Colibri VF50 é baseado no *System-On-Chip* (SoC; em português circuito integrado) *Freescale Vybrid*, o qual possui memória *DDR3 RAM* de 128MB e *NAND Flash* de 128MB. O SoC *Vybrid* possui um processador *Cortex-A5*, o qual suporta uma frequência de *clock* de até 500 MHz, bem como um microcontrolador *M4*, constituindo assim uma arquitetura heterogênea com dois núcleos[11].

O SoC VF50 oferece uma ampla gama de interfaces tais como *General Purpose Input/Output* (GPIO), padrão *Inter-Integrated Circuit* (I2C) canais de *Analog-to-Digital Converter* (ADC, em português conversor analógico-digital), dentre outras características. A Figura 1 representa o diagrama de blocos do módulo. O módulo possui pinagem *SODIMM* com 200 pinos, o qual é conectado a placa base Viola Plus[11].

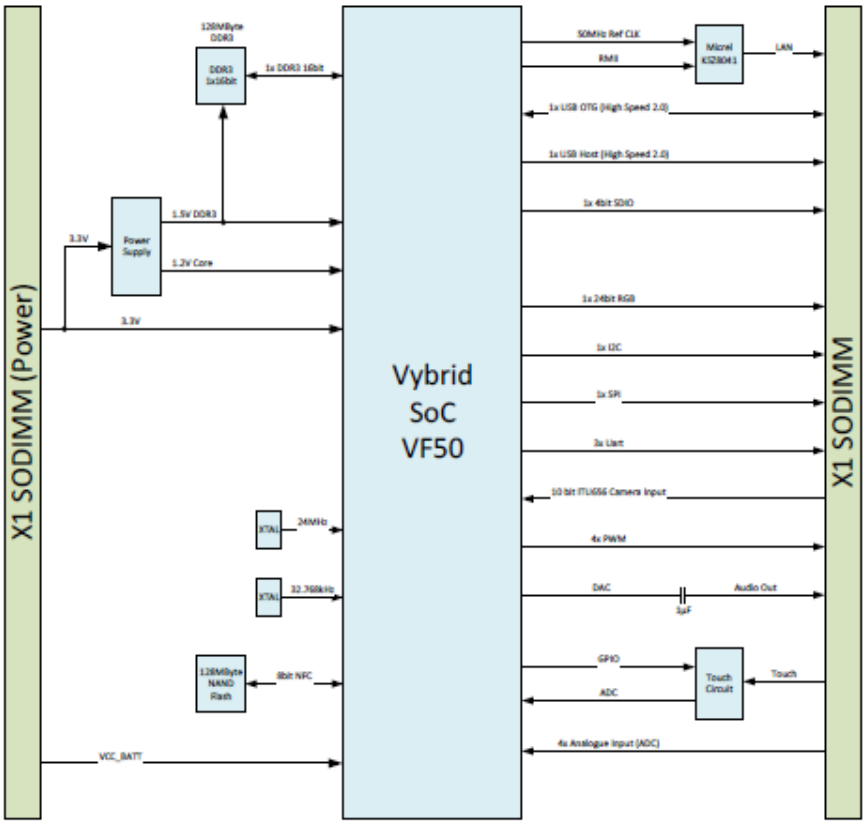


Figura 5: Diagrama de blocos do Colibri VF50[11]

3.1.1.2- Placa base Viola Plus

A placa base Viola Plus possui um fator de forma reduzido, característica esta devido a otimização de seu preço versus funções disponíveis. A Figura 2 mostra o diagrama de blocos da arquitetura da placa base e na Figura 3 os conectores presentes nesta placa [24]. A finalidade da mesma é auxiliar no desenvolvimento do produto, onde a mesma ao ser utilizada juntamente com SoMs compões uma solução *Single Board Computer* (SBC; em português Computador de placa única), porém a mesma pode ser substituída por uma placa concebida pelo desenvolvedor, que melhor se adeque as necessidades do produto a ser desenvolvido.

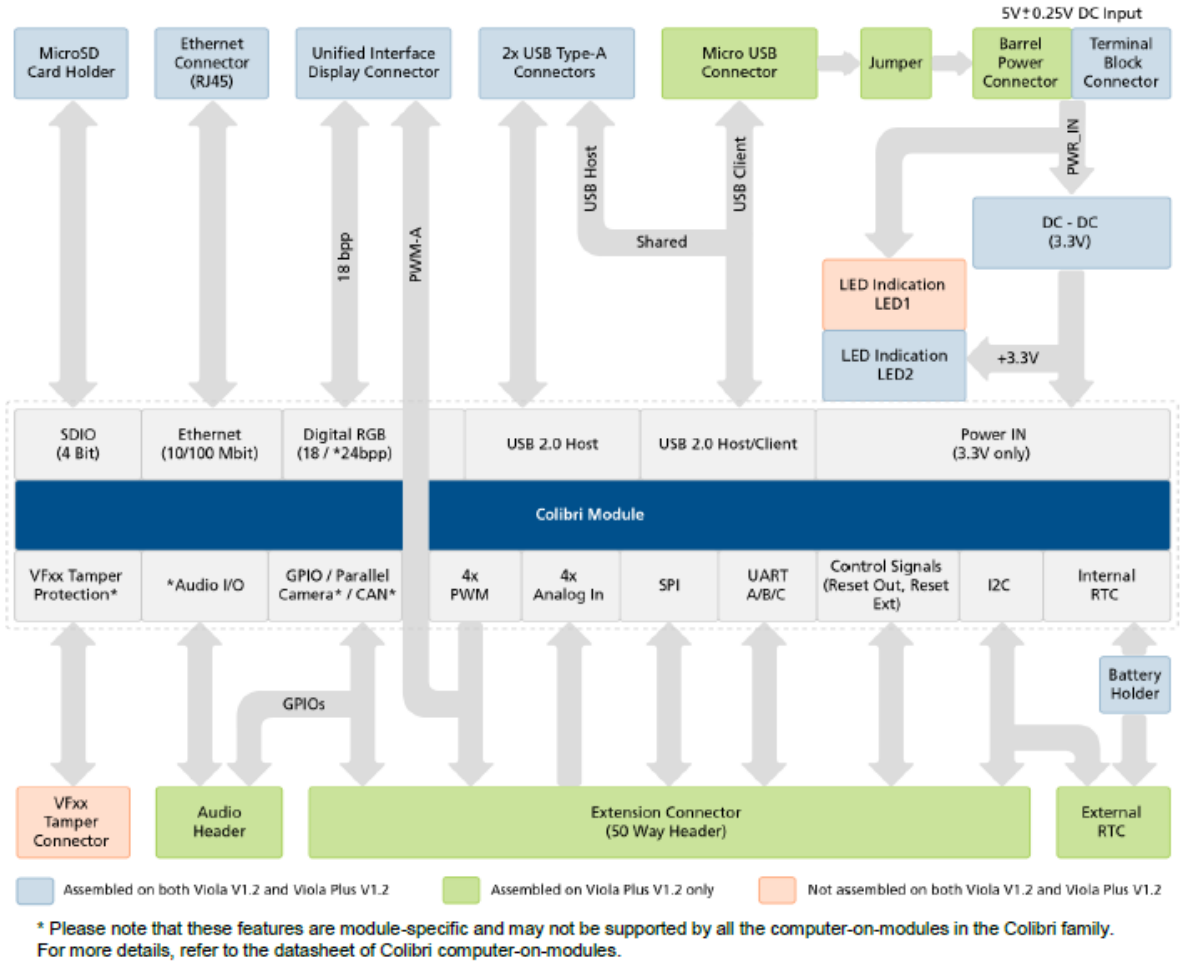


Figura 6: O diagrama de blocos que representa os principais subsistemas de hardware e interfaces disponíveis na Viola Plus [24].

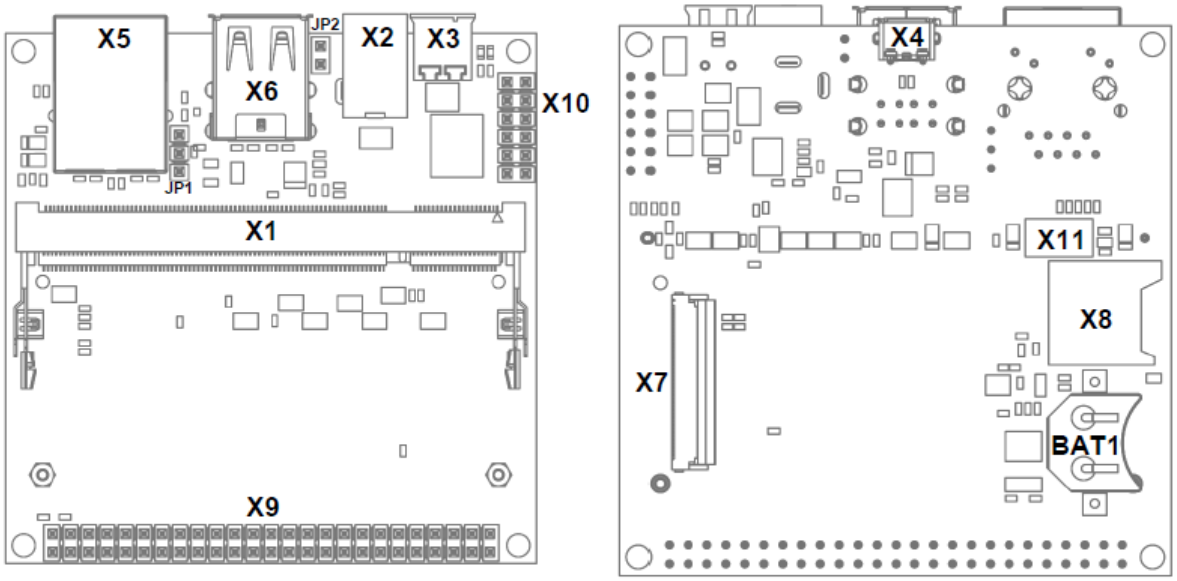


Figura 7: Conectores da base board Viola Plus [24]

X1 – Colibri SODIMM

X2 - Conector de alimentação

X3 - Conector de Alimentação

X4 - USB client

X5 - Conector Ethernet

X6 - 2xUSB Host,Type A

X7- Interface Unified TFT

X8 - Micro SD Card

X9 - Conector de extensão - Através deste conector estão as saídas de sinais como o de GPIO, I2C, UART, PWM e ADC.

X10 - Conector de Audio

X11 - VFxx Tamper Connector

3.1.1.2.1- Saídas utilizadas no projeto

X2- Conector de fonte de energia do módulo: Foi utilizada uma fonte capaz de fornecer 5V e 1A.

X3 - Este conector está conectado em paralelo ao conector X2, devido a esta característica o mesmo foi utilizado para alimentação dos sensores de corrente, cuja alimentação deve ser mantida em 5V.

X6 - Em uma das Portas USB foi conectado o *Dongle Wi-Fi LM006* utilizado para conexão com a internet

X9 - Neste conector foram utilizados os pinos do sinal ADC para leitura dos sensores de corrente

PIN 47 - Canal ADC0
PIN 48 - Canal ADC1
PIN 49 - Canal ADC2
PIN 50 - Canal ADC3

3.1.2 – *Microsoft Azure*

Devido as vantagens citadas acima, foi utilizado o *Microsoft Azure*, um dos vários serviços de computação em nuvem disponíveis, onde o mesmo oferece serviços de armazenamento, análise preditiva, criação de diferentes tipos de máquinas virtuais, dentre outros. Atualmente a plataforma *Azure* oferece cerca de 62 produtos em sua base de serviços, tendo suporte a diferentes linguagens de programação e plataformas, visando facilitar a migração de qualquer aplicação para a nuvem[17]. Os serviços desta plataforma utilizados serão detalhados a seguir.

3.1.2.1- IoT Hub

O *IoT Hub* do *Azure* é um serviço totalmente gerenciado que permite comunicações bidirecionais confiáveis e seguras entre milhões de dispositivos IoT e um *back-end* da solução. Isto é possível por meio das seguintes característica: Enviar e receber mensagens da nuvem para o dispositivo e vice-e-versa, protegendo esta comunicação utilizando credenciais de segurança e controle de acesso pelos dispositivos a nuvem; Gerenciar a identidade e conectividade do dispositivo através do monitoramento dos eventos descritos anteriormente; Possuir bibliotecas que suportam diferentes plataformas de diferentes dispositivos[18].

Este serviço é responsável por enviar os dados para a nuvem, o qual encapsula a mensagem no formato *JSON*, formato este de intercâmbio de dados baseado em um subconjunto de *JavaScript*, facilmente interpretado por humanos e formato direto para que os programas criem e analisem as mensagens[63]. As mensagens encapsuladas são enviadas até a plataforma do IoT Hub através do protocolo *Hypertext Transfer Protocol* (HTTP), o qual consiste em um protocolo de rede utilizado para entregar praticamente todos os arquivos e outros dados (chamados de recursos) na *World Wide Web*, sejam eles arquivos de imagem, resultados de consulta ou qualquer outra coisa[64]. Em seguida o a mensagem é verificada, se validado a segurança da mensagem, a mesma é armazenada por este serviço por até 7 dias[18].

3.1.2.2- *Stream Analytics*

O *Stream Analytics* , do *Azure*, é um mecanismo de processamento de eventos instantâneamente e totalmente gerenciado. Isto é possível por meio da configuração da computação analítica, instantânea, em dados de *Stream ing* de dispositivos, sensores, sites da *web*, aplicativos, sistemas de infra-estrutura, dentre outros, onde estes dados são manipulados utilizando uma linguagem semelhante ao *SQL*[19].

Uma das finalidades deste processamento é a identificação de situações, de risco ou não, instantaneamente, tais como identificação de fraudes, proteção de identidade e de dados, dentre outras situações, onde um sinal de alerta ou comando é enviado ao dispositivo ou sistema[20]. Outra função é a manipulação destes dados através de funções, tais como agrupamento de dados em uma determinada janela de tempo, utilização de funções aritméticas no tratamento dos dados, dentre outras, e em seguida encaminhando estes dados para outros serviços do *Azure* ou fora desta plataforma[19].

3.1.2.3- Power BI

Power BI consiste em um conjunto de ferramentas visuais de análise de negócios[21]. Este serviço tem como finalidade exibir de maneira gráfica e de fácil interpretação os dados que são provenientes de outros serviços, tais como *Azure, SQL, Excel*. Constituindo em uma ferramenta que consiste no *front-end* de uma solução, exibindo relatórios que permitam a atualização automática dos dados, bem como o acesso através de diferentes aplicativos, tais como tablet e smartphones.

3.1.3 – NodeJS

Node.js é um interpretador de código *JavaScript* que roda no servidor, que neste caso é a própria placa da *Toradex*. Sendo baseado no interpretador *V8 JavaScript Engine* (interpretador de JavaScript *open source* implementado pelo *Google* em C++ e utilizado pelo *Chrome*)[25].

Node JS foi escolhido devido a sua simplicidade, onde dispõe do mínimo de *APIs* tornando-o mais leve, porém para uma aplicação mais complexa, pode-se facilmente adicionar pacotes com demais funcionalidades.

Para elaboração do programa foi utilizado editor de texto *VI* e para a depuração do código foi utilizado a ferramenta *Nodemon*, que consiste em *file watcher* que roda internamente o próprio comando *node* [27].

3.1.4 - Sensor de Corrente ACS712

Para realização do projeto optou-se pela escolha do sensor ACS712 para medição de correntes de até 5A, devido a característica do projeto ser aplicado em um conjunto de tomadas, nas quais geralmente são conectados aparelhos que consomem baixa corrente.

Neste projeto utilizou-se um sensor do tipo invasivo, ou seja, há a necessidade de interromper o circuito para realização da medição de corrente, devido ao seu baixo custo e medição de correntes de baixo consumo. Tal escolha se deu em função de que os sensores não-invasivos disponíveis no mercado realizarem a medição de correntes maiores, de 20A ou mais, impossibilitando uma medição precisa para este projeto. Na Figura 4 é apresentado o diagrama de blocos do sensor ACS712.

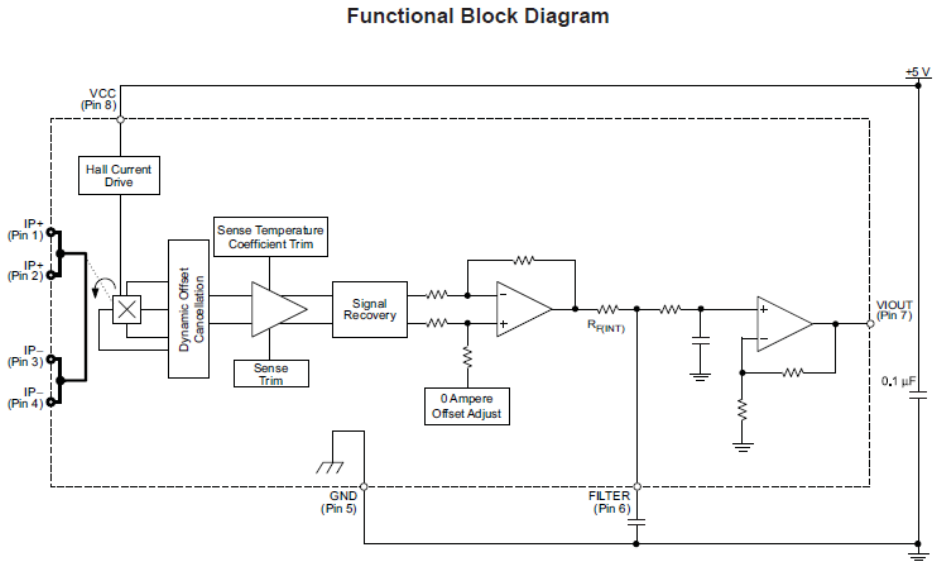


Figura 8: Diagrama de blocos funcional do sensor ACS712 [23]

3.1.5 - Módulo *Wi-Fi LM006*

Foi escolhido para este projeto um módulo *Wi-Fi LM006* devido ao mesmo proporcionar uma conectividade com a internet sem a dependência de um acesso físico, como o cabo ethernet.

O módulo LM006 é um adaptador USB 2.0 para estabelecer a conexão com a internet, o qual possui taxas de transferência de até 150 Mbps. Este adaptador foi utilizado por proporcionar um bom desempenho, com o mínimo consumo de energia e por ser de pequeno fator de forma [28].

3.2 - Métodos

Nesta seção serão descritos os métodos empregados neste projeto até o desenvolvimento do protótipo.

3.2.1 - Configuração do ambiente no módulo Colibri VF50

O ambiente utilizado no módulo foi *Linux Image V2.6* pré-compilada da *Toradex* [29], utilizando os passos descritos para a instalação da mesma, disponíveis no próprio site da *Toradex*[30]. Para o acesso ao módulo foi utilizado a placa Conversor *USB Serial FTDI FT232RL* e por meio do terminal do computador executada a seguinte instrução. Para o primeiro *login* o usuário é *root* e não possui senha, porém a mesma poderia ser acessada via SSH.

```
root@user:~# picocom -b 115200 /dev/ttyACM0
```

Para configuração de rede no módulo foi utilizado *LM006* [28], ao inseri-lo em uma porta *USB* (conector X6) o mesmo será identificado através da saída abaixo:

```
[ 151.035883] usb_phy_bringup_host_controller: timeout waiting for USB_PORTSC_PE
[ 151.195650] usb 2-1: new high speed USB device number 2 using tegra-ehci
[ 151.364040] usb 2-1: New USB device found, idVendor=148f, idProduct=3070
[ 151.370894] usb 2-1: New USB device strings: Mfr=1, Product=2, SerialNumber=3
[ 151.385573] usb 2-1: Product: 802.11 n WLAN
[ 151.389803] usb 2-1: Manufacturer: Ralink
[ 151.393842] usb 2-1: SerialNumber: 1.0
[ 151.545616] usb 2-1: reset high speed USB device number 2 using tegra-ehci
```

Para habilitar o *Wi-Fi* e identificação das redes disponíveis são utilizadas as instruções abaixo:

```
root@colibri-vf50:~# connmanctl enable wifi
root@colibri-vf50:~# connmanctl scan wifi
```

Em seguida para exibir as redes disponíveis foi utilizado o seguinte comando, onde foi exibido as redes conforme a saída abaixo:

```
root@colibri-vf50:~# connmanctl services

*AR Wired        { ethernet_00142d486cef_cable }
 <SSID>          { wifi_<HASH>_managed_psk }
```

Finalmente, com o propósito de ajuste da segurança WPA PSK, foi utilizada a configuração abaixo, onde no lugar de *Name* foi preenchido com o nome da rede *Wi-Fi* a ser conectada, obtida na instrução anterior, e em *Passphrase* foi preenchido com a senha da rede *Wi-Fi*:

```
root@colibri_vf50:~# vi /var/lib/connman/<SSID>-psk.config

[service_wifi_<HASH>_managed_psk]
Type = wifi
Name = <SSID>
Passphrase = <PASSPHRASE>
```

Por fim executar o comando abaixo para conectar na rede, obtendo a saída logo abaixo da instrução, a qual confirma a conectividade com a rede:

```
root@colibri_vf50:~# connmanctl connect wifi_<HASH>_managed_psk

[ 1165.219739] ADDRCONF(NETDEV_CHANGE): wlan0: link becomes ready Connected
```

O passo seguinte consistiu na instalação dos pacotes necessários para o desenvolvimento da solução, que consistem no gerenciador de pacotes NPM, Nodemon, NodeJS. As instruções necessárias para a instalação foram descritas abaixo:

```
root@colibri_vf50:~#opkg update
root@colibri_vf50:~#opkg install nodejs-npm
root@colibri_vf50:~#npm install -g nodemon
```

3.2.2- Corrente medida

A leitura dos sensores é feita por meio das entradas analógicas disponíveis no módulo VF50. Esta leitura é feita por meio do arquivo *sysfs*, o qual envolve somente a leitura de um arquivo[31]. Abaixo encontram-se os canais ADC utilizados para leitura e os respectivos arquivos de acesso aos mesmos.

Tabela 1: Pinos e arquivos utilizados para leitura do canal ADC[31]

Canal	PIN Viola	Arquivo
AD0	X47	/sys/bus/iio/devices/iio:device0/in_voltage8_raw
AD1	X48	/sys/bus/iio/devices/iio:device1/in_voltage9_raw
AD2	X49	/sys/bus/iio/devices/iio:device0/in_voltage8_raw
AD3	X50	/sys/bus/iio/devices/iio:device1/in_voltage9_raw

Após a leitura de cada canal fez-se necessária a conversão dos valores lidos de tensão para conversão em corrente *RMS*. Para tal finalidade lança-se mão da manipulação destes dados e armazenamento em vetores. Para as operações que envolvem a utilização de vetores, primeiramente foi necessária a instalação do pacote abaixo.

```
root@colibri_vf50:~#npm install vectors
```

Uma vez lidos os valores e armazenados em um vetor, faz-se necessário encontrar os valores máximos e mínimos desta leitura, como exemplificado abaixo para ADC0.

```
maxValue=0;
minValue=4096;
for(i=0;i<100;i++){
 V_aux[i]= fs.readFileSync('/sys/bus/iio/devices/iio:device0/in_voltage8'_raw');
  if (V_aux[i]<minValue){
    minValue=V_aux[i];
  }
  if(V_aux[i]>maxValue){
    maxValue=V_aux[i];
  }
 }
```

Após obtidos os valores máximos e mínimos realizou-se o cálculo do valor de pico da tensão, a tensão RMS (Vrms) e subsequentemente a corrente RMS (Arms). Para isto inicialmente foi calculado a resolução do conversor, onde dividiu-se a tensão de referência do canal ADC pelo número de níveis de quantização do módulo VF50. Em seguida para o cálculo da amplitude do sinal foi calculado o Va multiplicando-se a resolução do conversor pela diferença de Vmax-Vmin lidos na saída do canal ADC, obtendo-se as seguintes fórmulas:

$$Resolucao\_Conversor=(Nível\_de\_Tensão\_Referencia\_ADC)/(Numero\_de\_Niveis\_Quantizacao\_ADC)$$

$$Vp=(Vmax\text{-}Vmin)*(Resolucao\_do\_Conversor)$$

Onde o $Nível\_de\_Tensão\_Referencia\_ADC = 3.3V$, consistindo na tensão de entrada do pino ADC, e $Numero\_de\_Niveis\_Quantizacao\_ADC = 4096$, onde o valor da resolução do módulo: 12 bits.

Em seguida foi calculado a tensão RMS lida por meio da fómula $Vrms=Vm/\sqrt{2}$, onde obteve-se a tensão média do sinal através da divisão da amplitude do sinal por 2, resultando na seguinte fórmula

$$Vrms= (Vp)/(2*\sqrt{2})$$

E para calcular a corrente RMS do conversor, dividiu-se tensão Vrms calculada anteriormente pela sensibilidade da saída do conversor, como descrita na fórmula abaixo, onde *Volts_por_Unidade = 0.185*, sendo a sensibilidade do sensor de corrente de 185mV/A

Arms=(Vrms) / (Volts_Por_Unidade)

3.2.3- IoT Hub

Após implementado o algoritmo de leitura dos dados dos sensores de corrente a etapa seguinte consistiu na configuração dos serviços do *Azure*. Primeiramente foi necessária a configuração do *IoT Hub*, serviço este responsável pelo envio e recebimento dos dados na plataforma *Azure*.

Para criação do *IoT Hub* foi necessário, primeiramente, criar uma conta no *Microsoft Azure*[32], a qual possui também uma versão *Trial* de 30 dias. Após criada a conta no *Microsoft Azure* passou-se para etapa de criação do IoT Hub. Na página inicial do *Azure* clicar em ***New>Internet of things> IoT Hub***, como ilustrado na Figura 9.

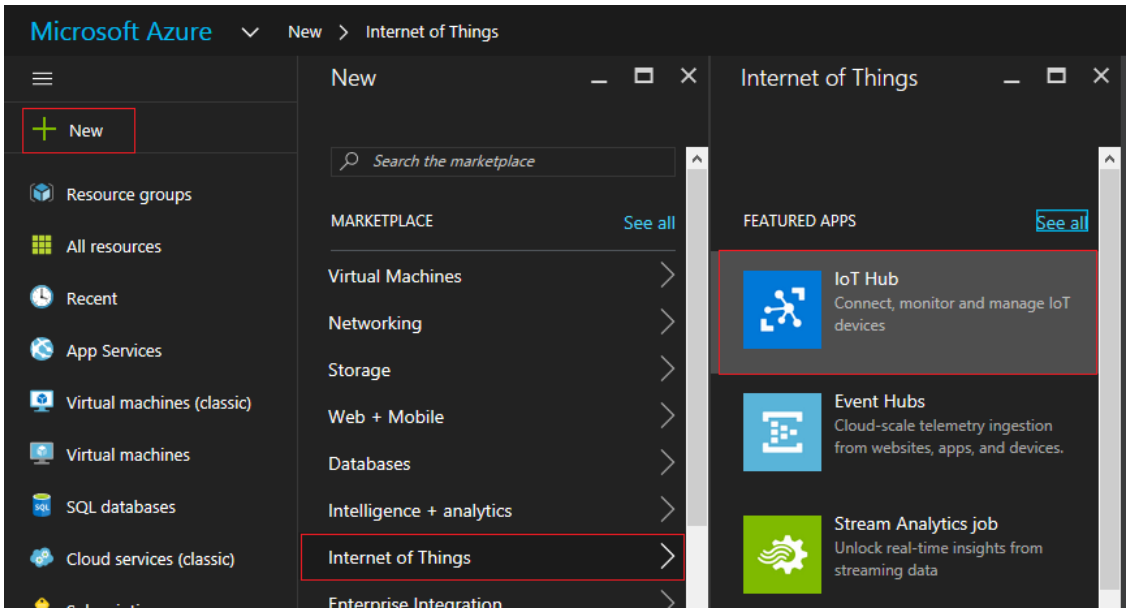


Figura 9: Página inicial do Azure[33]

A etapa seguinte consistiu na escolha das configurações do IoT Hub [34], abaixo segue os campos necessários para configuração do mesmo.

- *Name:* Preencher com o nome para IoT Hub.

- *Pricing and scale tier:* Escolher o tipo de *IoT Hub*, isto ficará a critério do uso estimado para o mesmo, neste projeto foi escolhida uma conta *Free*, pois a quantidade de dados enviados para o mesmo é baixa.
- *Resource Group:* Ao criar um grupo de recursos, todos os serviços tais como *Virtual Machine*, banco de dados, dentre outros, serão implantados, gerenciados e monitorados como recursos da sua solução como um grupo. Por isso é indicado criar um grupo de recursos para gerenciá-los [35].
- *Location:* selecionar a localização para armazenar o *IoT Hub*, de preferência escolher a localização mais próxima, para diminuir assim a latência do serviço.

Por fim clicar em ***Create*** para que o IoT Hub seja criado, como ilustrado na Figura 10.

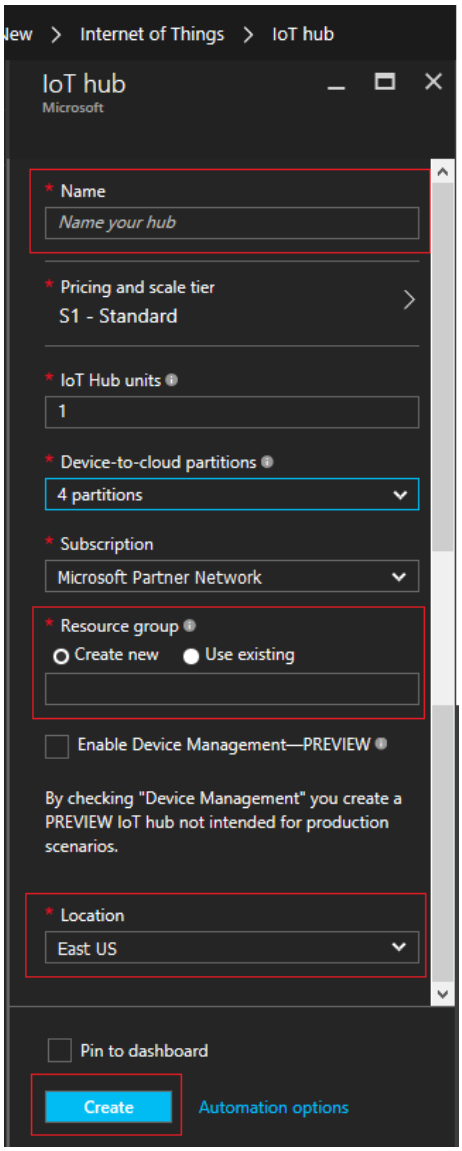


Figura 10: Configuração IoT Hub [36]

Após a criação da conta no IoT Hub foi necessário adicionar cada dispositivo a ser conectado ao *IoT Hub* por meio da ferramenta *iothub-explorer*. Esta etapa foi realizada por meio do terminal do computador executando a seguinte instrução:

```
root@user:~#npm install iothub-explorer@latest
```

Para adicionar o dispositivo que enviaram os dados para a nuvem, foi necessário, na tela inicial do *IoT Hub* recém-criado, selecionar a opção ***Settings>Shared access polices>iothubowner*** , conforme ilustrado na Figura 11. Foi necessário copiar a *Connection String-primary key*, pois a mesma será utilizada no próximo passo, para criar o dispositivo e assim gerar a chave para enviar a mensagem.

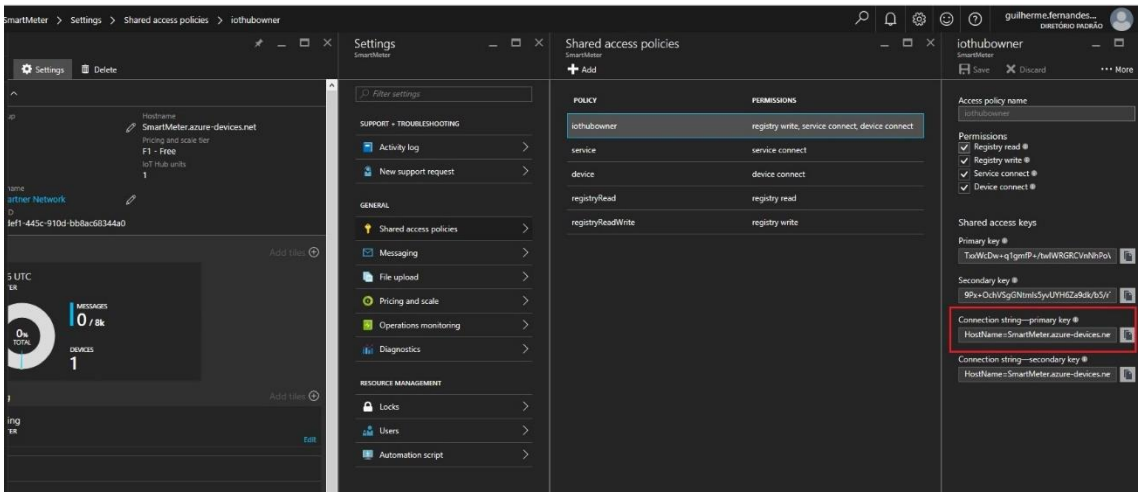


Figura 11: Consulta conection string [37]

Para cadastrar o dispositivo no *iothub-explorer* foi utilizada a instrução abaixo, substituindo o campo *iothub-explorer* pela *connection string* obtida anteriormente, em seguida, escolheu-se o nome para o dispositivo, que neste projeto foi utilizado como *smartmeter2* e, por fim, a opção *--connection-string* exibirá a cadeia de conexão do dispositivo, como ilustrado na Figura 12.

```
root@colibri_vf50:~#iothub-explorer "HostName=toradex.azure-
devices.net;SharedAccessKeyName=iothubowner;SharedAccessKey=putyoursharedaccesskeyfromthe
connectionstringhere" create smartmeter2 --connection-string
```

Caso seja utilizado mais de um dispositivo para enviar dados, deverá ser utilizada a mesma instrução acima, alterando-se somente o nome de cada dispositivo. Neste caso será criada para cada dispositivo uma cadeia de conexão do dispositivo diferente, e cada dispositivo deverá adicionar a respectiva cadeia de conexão ao programa que enviará a mensagem. Na Figura 12 encontra-se o exemplo da saída desta instrução, ao final será criada uma *ConnectionString*, sendo esta a cadeia de conexão do dispositivo.

Figura 12: Exemplo de saída de instrução de uma cadeia de dispositivo [38]

Outro modo para consulta dos dispositivos criados é através do IoT Hub, na opção *Devices*. Nesta aba foram exibidos os dispositivos cadastrados e ao clicar em cada dispositivo abrirá outra aba chamada *Devices Details*, onde se encontra a *connection string* de cada dispositivo, ou seja a cadeia de conexão do dispositivo, conforme a Figura 13.

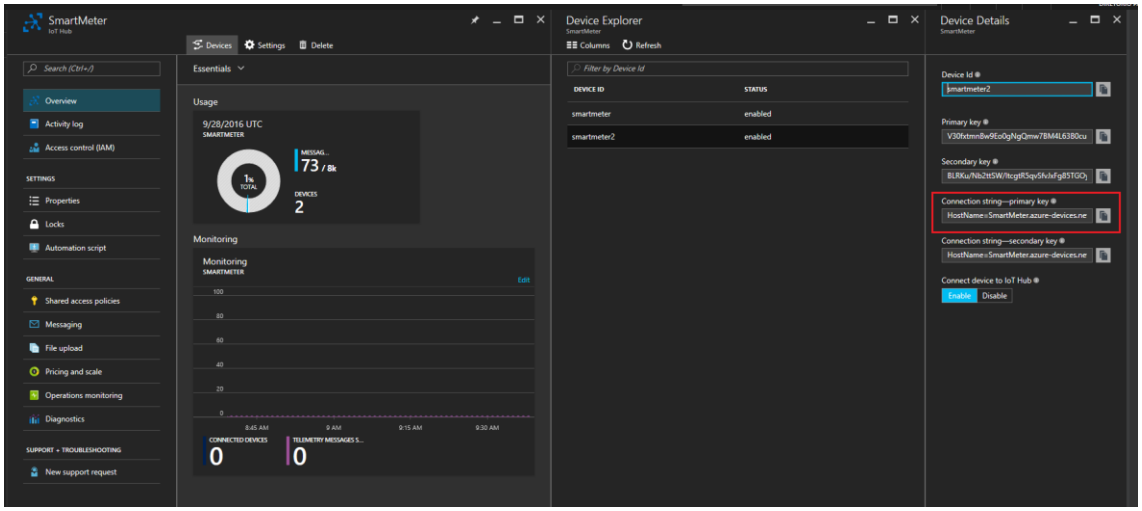


Figura 13: Consulta à cadeia de conexão do dispositivo através do IoT Hub [39]

Esta cadeia de conexão do dispositivo foi utilizada como uma chave para a autenticação de cada dispositivo, quando a mensagem é recebida no *IoT Hub*. Caso esta chave seja válida, a mensagem será recebida e processada na plataforma do *Azure*, caso contrário será gerada uma mensagem de erro em seu código, indicando que a mensagem não foi entregue.

Em seguida para enviar a mensagem foi necessária a instalação da *SDK* do *Azure* no módulo utilizando as seguintes instruções:

```
root@colibri_vf50:~#npm install -g azure-iot-device@1.0.1
root@colibri_vf50:~#npm install -g azure-iot-device-http@1.0.1
```

Para a criação do programa responsável pelo envio dos dados para a nuvem, a etapa inicial do programa consistiu na declaração da variável *Connection String*[40], esta variável deverá conter a cadeia de conexão do dispositivo anteriormente descrita, como segue a seguir:

```
var connectionString = "HostName=toradex.azure-
devices.net;DeviceId=smartmeter;SharedAccessKe=somesharedaccesskeyreturned"
```

Em seguida são declaradas as principais funções do programa, as quais são responsáveis pelo envio das mensagens para o IoT Hub:

```
sendInterval.handlerGet = setInterval(getAllSensors, sendInterval.timerGet);
sendInterval.handlerSend = setInterval(sendToIotHub, sendInterval.timerSend);
```

A primeira função, *sendInterval.handlerGet = setInterval(),* fez uma chamada à função *getAllSensors(),* a qual armazena as leituras de corrente ao fazer uma chamada a função *rdADC( )*. Em seguida foi realizado o cálculo da potência em cada sensor, através da multiplicação o valor da corrente pela tensão da tomada. Esta função encontra-se no Apêndice deste trabalho (Algoritmo 1)[40].

Esta segunda função, *sendInterval.handlerSend = setInterval(),* irá realizar uma chamada a função *sendToIotHub(),* a qual encapsula os dados obtidos das leituras dos sensores em uma string no formato *JSON,* que em seguida enviou esta mensagem para o IoT Hub[40].

Para implementação do código descrito acima foi utilizado o editor de texto VI, o arquivo SendData.js foi criado e posteriormente editado através do comando abaixo:

```
root@colibri_vf50:~# vi SendData.js
```

Para execução do programa foi utilizado o comando *nodemon* abaixo. Se todos os processos não resultarem nenhum erro, será exibida a mensagem que confirmará o envio para IoT Hub e exibidos os dados encapsulados no formato *JSON*, caso contrário será exibida uma mensagem de erro na tela.

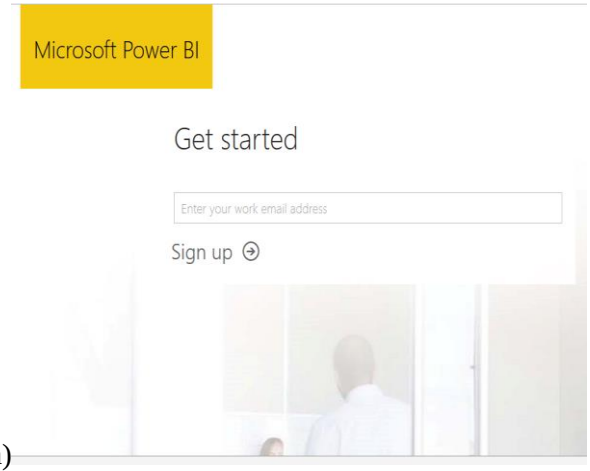
```
root@colibri_vf50:~#nodemon SendData.js
```

Para o monitoramento dos dados enviados, foi executada a seguinte instrução por meio do terminal do computador, substituindo *your_iothub_connection_string* pela a cadeia de conexão do dispositivo, e *your device* pelo nome criado para o seu dispositivo [40].

```
root@colibri_vf50:~# iothub-explorer "your_iothub_connection_string" monitor-events yourdevice
```

3.2.4 – Criar conta no Power BI

Este capítulo aborda a criação da conta no Power BI, devido a necessidade da utilização desta ferramenta na configuração da saída do *Stream Analytics* [41]. Para a criação da conta será solicitado o e-mail para a criação da mesma, como ilustrado na Figura 14a. Em seguida foi enviado um e-mail de confirmação para a conta cadastrada, após esta confirmação foi redirecionado a uma página para configuração da conta, como mostrada na Figura 14b, em seguida foram preenchidos com nome e senha criados para esta conta.

a) 

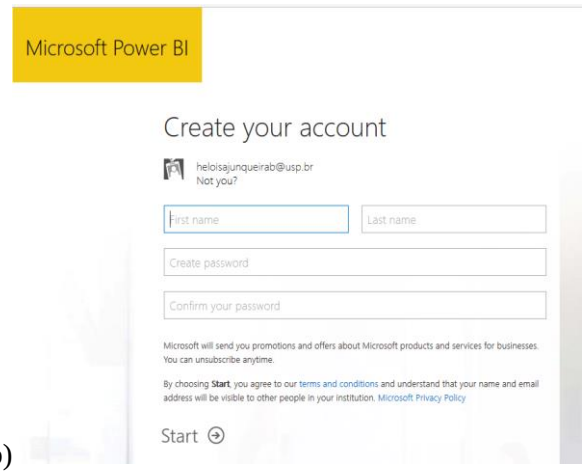
b) 

Figura 14: a) Página inicial para criação da conta b)Página para configuração da conta [42]

Para maior segurança foi solicitado ou o número de telefone ou um email, diferente do cadastrado, para o envio de um código de verificação, como mostrado na Figura 15. E então foi criada a conta no Power BI, sendo exibidos alguns temas que podem ser utilizados no mesmo, como mostrado na Figura 16.

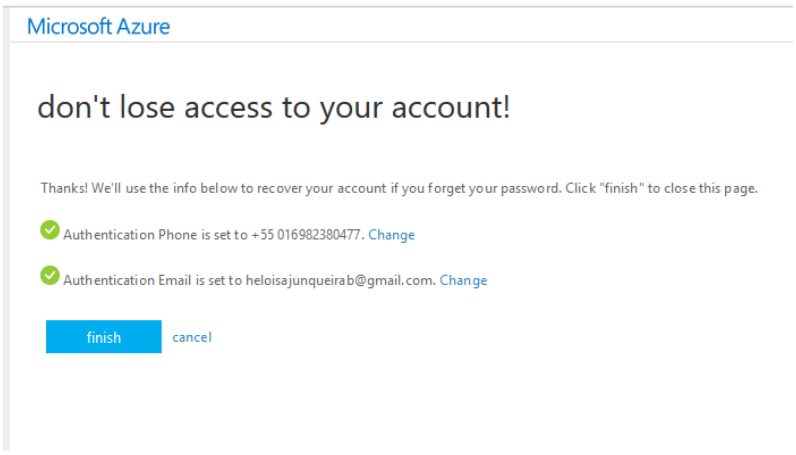


Figura 15: Solicitação de email ou telefone para verificação da conta [43]

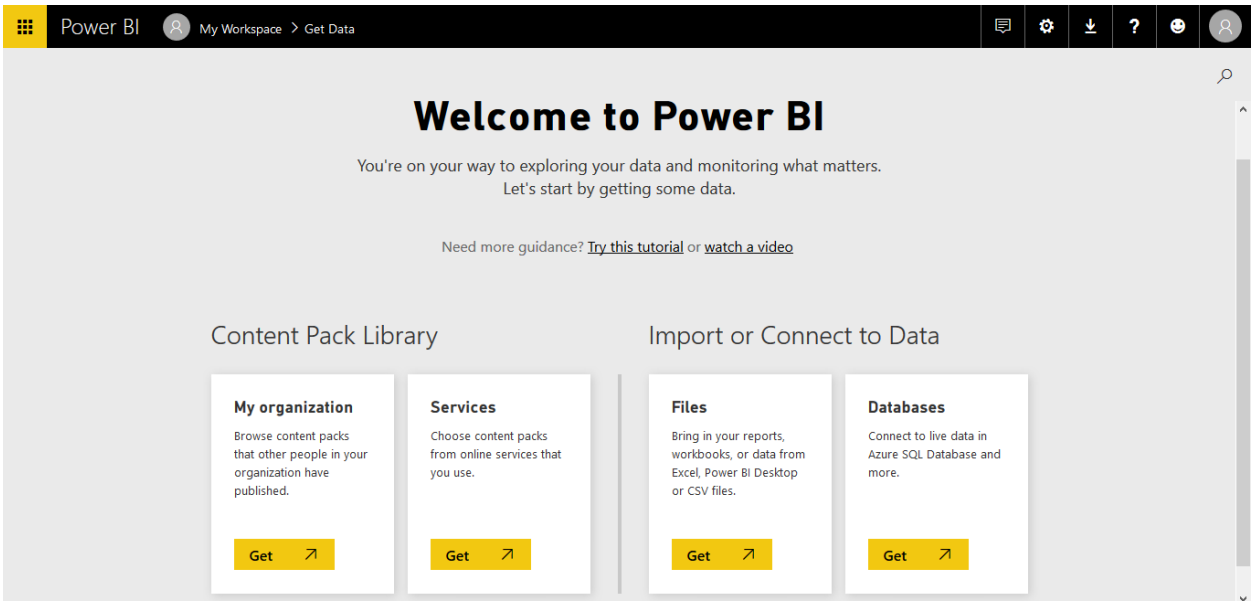


Figura 16: Tela inicial do Power BI [44]

3.2.5 - *Stream Analytics*

Após a configuração da etapa de leitura e envio dos dados para a nuvem, será abordado neste capítulo o processamento destes dados na nuvem. Para tal finalidade foi utilizada a ferramenta *Stream Analytics* da plataforma *Azure*.

Para a criação deste serviço será necessário entrar na página inicial da conta criada no *Microsoft Azure*, em seguida clicar na opção ***New>Internet of Things>Stream Analytics Job***, como mostrado na Figura 17-a. Em seguida, na aba *New Stream Analytics Job* será realizada a configuração do serviço do *Stream Analytics* , o qual deverá ser preenchido o campo *Job Name* com o nome a ser criado para o *Stream*

Analytics , os campos *Resource Group* e *Location* deverão ser os mesmos escolhidos para o IoT Hub criados anteriormente, como mostrado na Figura 17-b, em seguida clicar em ***Create*** e o *Stream* Analitics será criado.

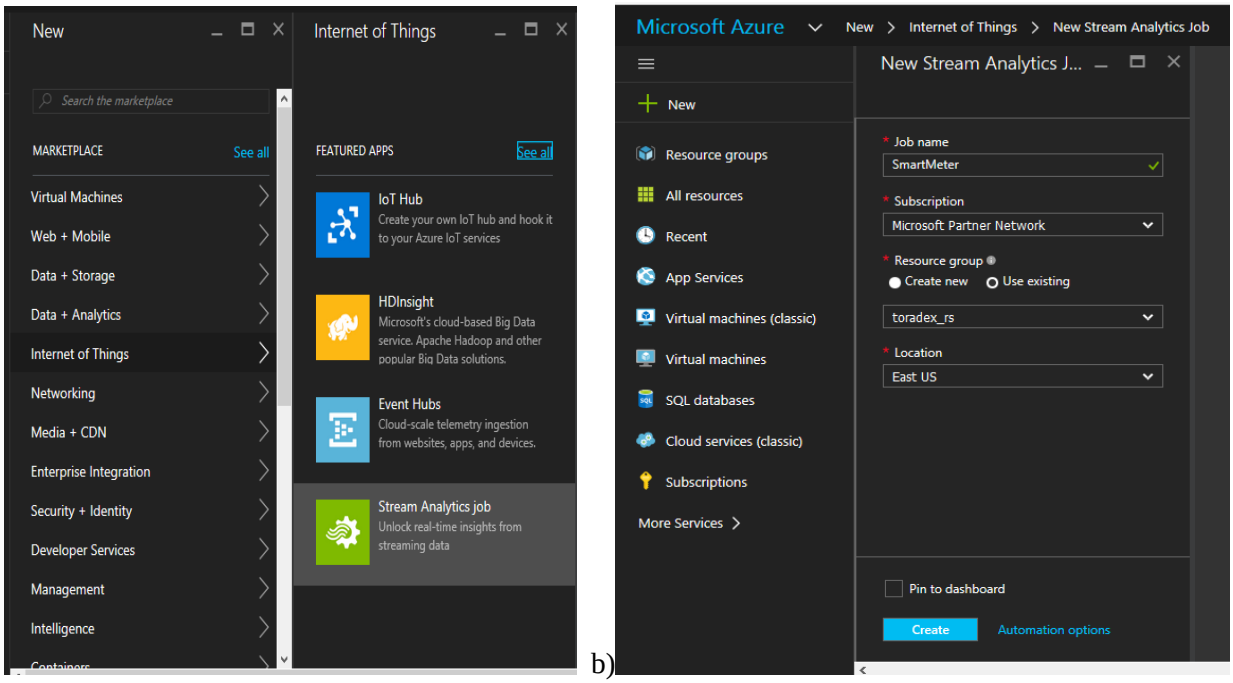


Figura 17: a) Criar *Stream Analytics* ; b) Configurar *Stream Analytics* [45]

Os passos para a configuração deste serviço consistem em: especificar a fonte de entrada dos dados que são enviados para a nuvem; configurar a saída para os resultados de seu trabalho; tratar os dados utilizando a linguagem SQL-like.

Inicialmente para configurar a entrada do serviço do *Stream Analytics* na tela inicial, será necessário seguir os passos descritos abaixo, como mostrada na Figura 18.

1) Clicar em ***Inputs****;*
2) Em seguida clicar em ***+Add****;*
3) Abrirá uma aba *New input*, em seguida preencher os seguintes campos:
   3.1) *Input Alias* com o nome a ser criado para esta entrada;
   3.2) *Source Type* definir como *Data Stream* ;
   3.3) *Source* definir como *IoT Hub*;
   3.4) *IoT Hub* selecionar o IoT Hub criado anteriormente;
   3.5) *Shared access policy name* escolher *iothubowner* ;

3.6) *Shared access policy key*, caso não seja preenchido automaticamente, preencher com a chave gerada na criação do IoT Hub no capítulo anterior, a cadeia de conexão do IoT Hub;

3.7) Os demais campos podem ser configurados com os valores padrão;

4) Clicar em ***Create*** e a entrada será criada, a qual receberá os dados provenientes do IoT Hub.

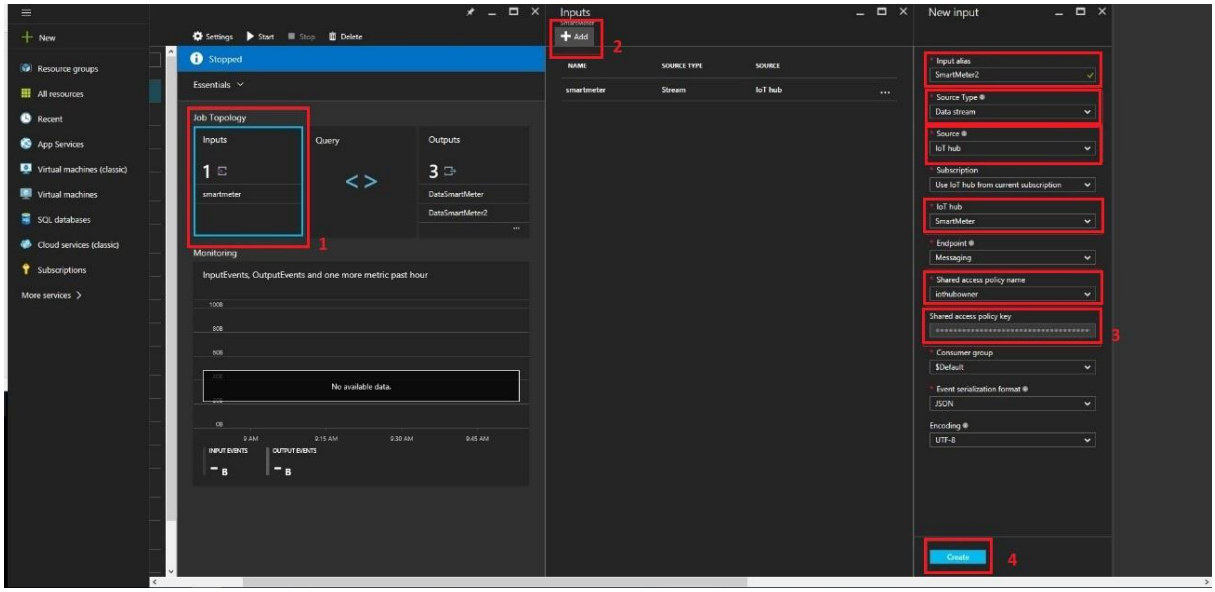


Figura 18 – Adicionar entradas no *Stream Analytics* [46]

Em seguida para configurar a saída do *Stream Analytics* , nesta mesma tela onde foi criada a entrada, conforme ilustrado na Figura 19, e seguir os passos descritos abaixo:

1) Clicar em ***Outputs***
2) Em seguida em ***+Add***,
3) Abrirá uma aba *New output*, em seguida preencher os seguintes campos:
    3.1) *Output Alias* preencher com o nome a ser criado para a saída;
    3.2) *Sink* escolher a opção *PowerBI*. Em seguida será solicitado preencher com o usuário e senha da conta criada do Power BI no capítulo anterior;
    3.3) *Group Workspace* selecionar a opção *My Workspace;*
    3.4)Nos campos *Dataset Name* e *Table Name* preencher com nomes de sua preferência, estes campos serão o nome do conjunto de dados que serão exibidos no Power BI

4) Clicar em ***Save*** e a saída será criada aba *Outputs*

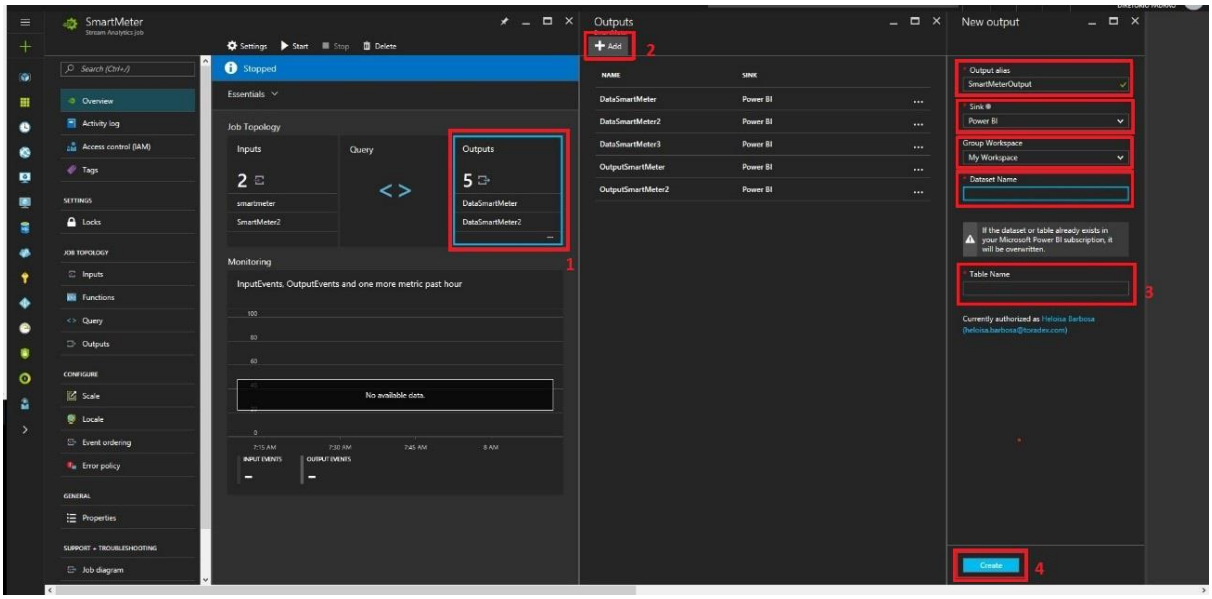


Figura 19: Adicionar saídas no *Stream Analytics* [47]

Após definidas as entradas e saídas do *Stream Analytics* , a etapa seguinte consiste na configuração do tratamento destes dados através da *Query* . Esta ferramenta realiza tratamento e manipulação dos dados, quando os mesmos são enviados em grande quantidade ou provenientes de milhares de dispositivos. Isto se torna possível através de ferramentas tais como com funções para agrupar, calcular a média de diferentes dados provenientes de uma janela de tempo, dentre outras.

A *Query* , neste projeto, recebe os dados de corrente e potência instantânea, provenientes do IoT Hub. Para uma melhor manipulação dos dados, estes foram separados em 3 diferentes saídas, onde a primeira envia somente dados de corrente, a segunda dados de potência por sensor e a terceira dados de potência total, onde esta última consiste no somatório da potência de todos os sensores. As funções utilizadas na *Query* foram retiradas da documentação *Stream Analytics Query* Language Reference [8]

Para a saída utilizando dados de corrente não foi utilizada nenhuma função para o tratamento dos dados, uma vez que a finalidade do mesmo é a exibição das correntes instantâneas.

Na saída que exibe os dados de potência consumida por sensor, utilizou-se a soma das potências lidas através da função *SUM()* e multiplicadas por 3, sendo esta a janela de tempo utilizada nas medições. Para agrupar estes dados em uma janela de 1 hora, para obtenção do consumo em KWh, foi utilizada a função *TumblingWindow().* Os cálculos foram realizados para saída de cada sensor e em seguida por cômodos da residência, ilustrando uma aplicação onde cada sensor é utilizado para medição de diferentes cômodos de uma residência e simulando a potência medida em uma hora como sendo a potência medida durante o mês, para simulação da potência mensal consumida por cômodo de uma casa.

Na terceira saída, para obtenção da potência total consumida por todos os sensores, o cálculo foi realizado somando-se as potências totais de cada sensor obtidas na saída anterior, simulando a potência medida em uma hora como sendo a potência medida durante o mês, para simulação da potência mensal total consumida de uma casa. Abaixo encontra-se a Figura 20, a qual ilustra a *Query* descrita acima.

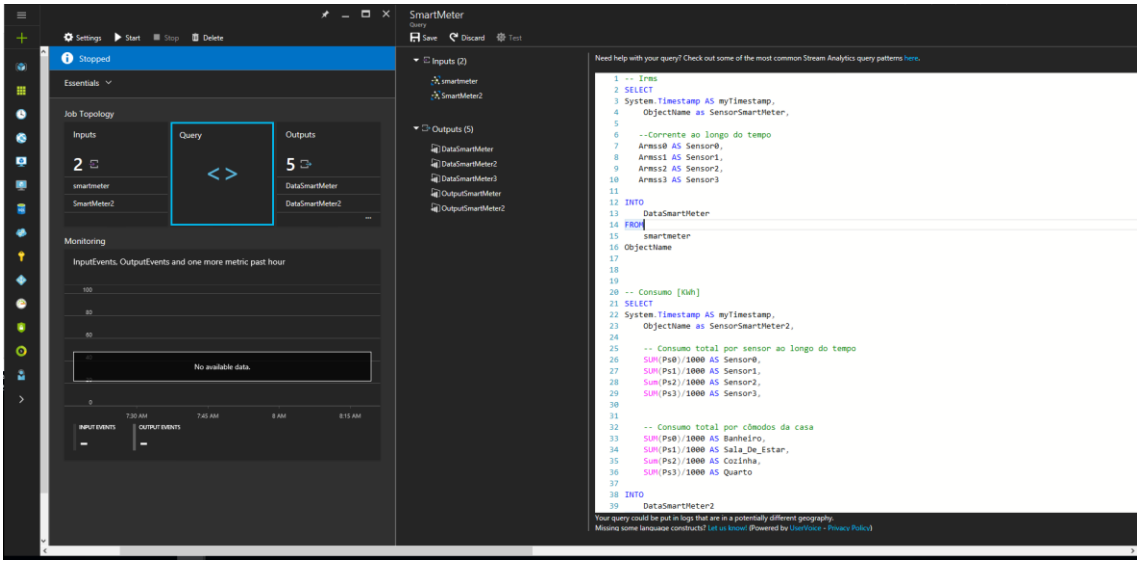


Figura 20: *Query Stream Analytics* [48]

Com o *Stream Analytics* configurado, através da entrada sendo o IoT Hub e a saída sendo o Power BI, pode ser iniciado o serviço através da opção ***Start***, na tela inicial do *Stream Analytics* , como mostrado na Figura 21a. Se o serviço foi configurado corretamente e os sensores estão enviando os dados para o IoT Hub, o serviço dentro de alguns minutos será iniciado. O mesmo poderá ser monitorado através da tela inicial do *Stream Analytics* , logo abaixo dos campos Input e Output, onde há um gráfico chamado ***Monitoring*** que exibe os dados sendo enviados para o *Stream Analytics* , bem como a saída destes dados para o Power BI, como mostrada na Figura 21b.

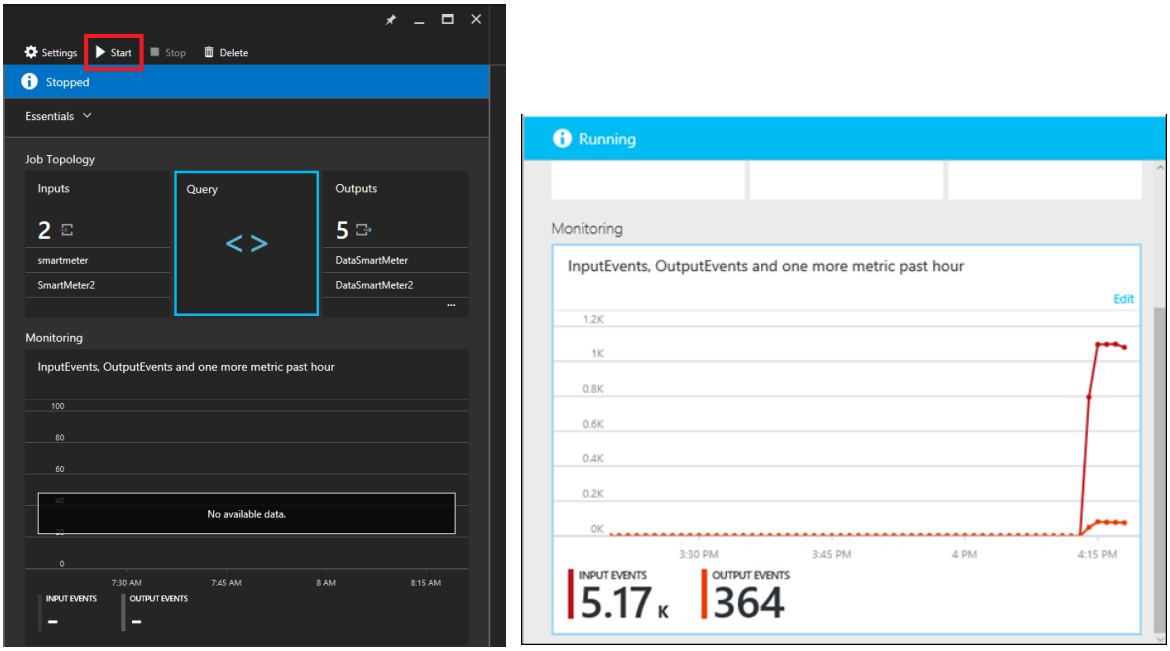


Figura 21: a)Iniciando serviço *Stream Analytics* b)gráfico de monitoramento *Stream Analytics* [49]

3.2.6 - Power BI

Após o serviço do *Stream Analytics* ser iniciado e estar enviando os dados para o Power BI, a etapa seguinte consiste na configuração dos gráficos através do Power BI, os passos para a configuração deste serviço será descrito abaixo:

1) Na tela inicial do Power BI no campo *Conjunto de dados*, serão exibidas as saídas de dados configuradas no *Stream Analytics* .
2) Ao clicar em cada saída do *Conjunto de dados* aparecerá no canto direito, a aba *Campos*, os dados configurados como saídas na *Query* .
3) Ao lado esquerdo dessa aba *Campos*, na aba *Visualizações*, estão contidos os tipos de gráficos a serem utilizados. Selecionar o tipo gráfico e em seguida os dados utilizados
4) Irá ser exibida as opções para a configuração dos valores, os quais variam conforme o gráfico escolhido, bem como os filtros a serem aplicados aos dados, que também variam conforme o tipo de gráfico.

A seguir será detalhado os passos para a criação do gráfico que mede a potência total dos sensores ao longo do tempo [50], os passos para os demais gráficos a serem criado serão semelhantes aos descritos abaixo:

4.1) Primeiramente, em *Conjunto de dados* clicar em PotTotal, uma das saídas que foram criadas no *Stream Analytics* .

4.2) Em seguida em *Visualizações* clicar em *Gráfico de colunas agrupadas*, na aba *Campos* foram selecionados os dados do somatório das potências *potencia_total*, criados anteriormente na *Query* , em seguida foi selecionado *mytimestamp*, medida de tempo dos dados enviados. Neste caso o Power BI já selecionou automaticamente o Eixo e Valores correspondentes

4.3) Logo abaixo, em filtros, caso seja necessário poder-se-ia aplicar alguns filtros, nos dados e no tempo, porém neste exemplo não houve a necessidade de aplicação dos mesmos.

4.4) Logo abaixo dos gráficos existe uma aba, cuja figura representativa é um rolo de tinta, o qual consiste no campo *formato* onde os gráficos podem ser personalizados através do preenchimento de campos tais como título, cores dos gráficos, dentre outros.

Todas as etapas descritas acima podem ser visualizadas através da Figura 22 abaixo.

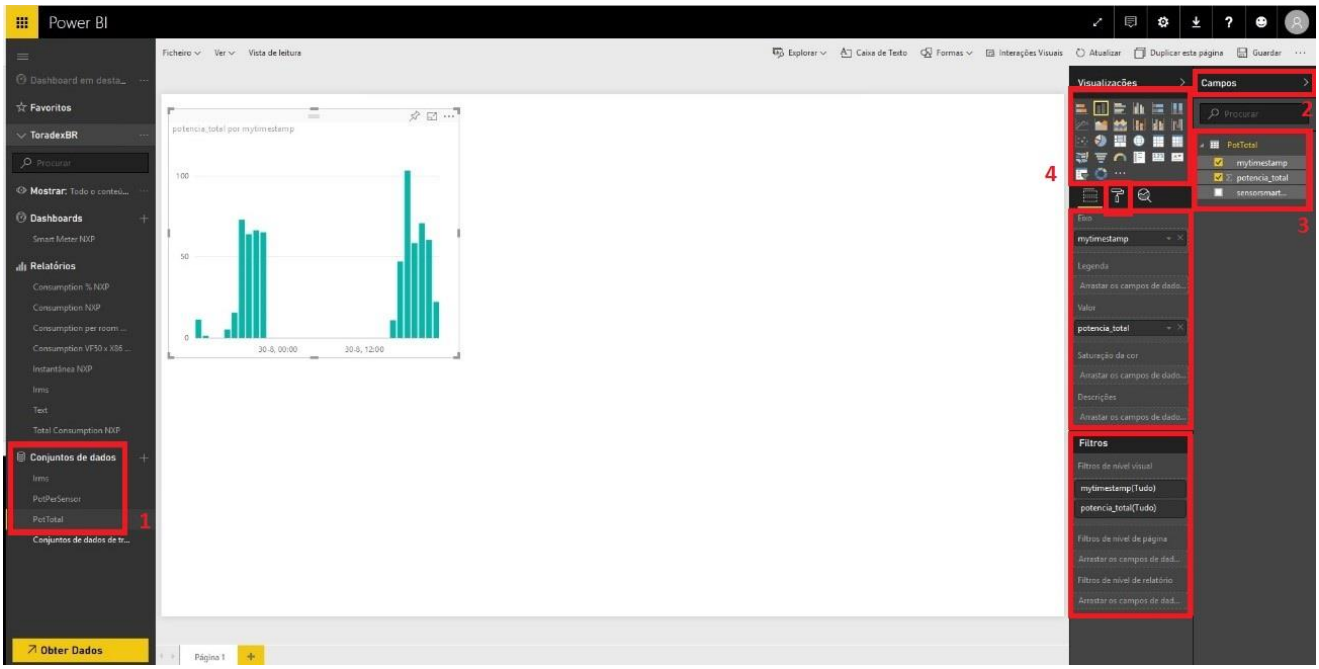


Figura 22: Campos para criação de um gráfico [51]

Após a customização deste gráfico será necessário salvá-lo como um relatório. Para isto clicar na opção ***File> Save As***, como mostrado na Figura 23a. Em seguida será exibida uma janela para criar o nome do relatório, e posteriormente clicar em ***Save***, como mostrado na Figura 23b. Após esta etapa será criado, canto esquerdo logo abaixo de relatório, o relatório recém criado.

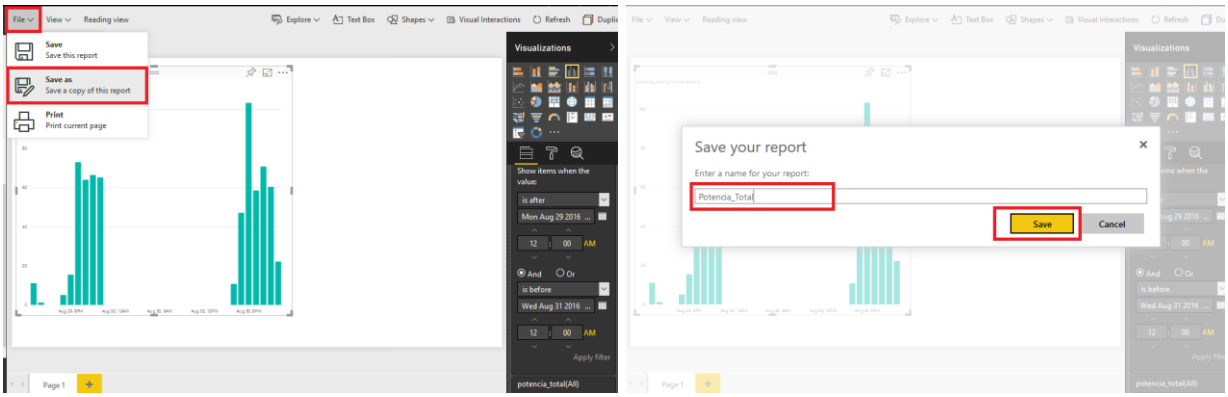


Figura 23: a) Salvar um relatório b) Janela para criação do relatório[52]

Ao criar um relatório, o gráfico somente será atualizado ao clicar na opção atualizar, ou seja, só irá atualizar de forma manual. Para que estes gráficos sejam atualizados automaticamente em tempo real será necessário fixa-los em um Dashboard. Para a criação do mesmo, em frente a opção Dashboard no canto esquerdo do PowerBI, clique no botão ***Add*** **+**, e em seguida preencha o nome para o seu Dashboard, como ilustrado na Figura 24a. Para adicionar o gráfico, criado anteriormente, no Dashboard recém criado, adicione cada gráfico por vez, para isto clicar em cima do gráfico e no canto superior direto aparecerá a figura de um alfinete, a opção ***Pin visual***, como ilustrada na Figura 24b. Ao clicar no mesmo aparecerá a janela ***Pin to dashboard***, então escolha a opção *Existing Dashboard*, selecione o Dashboard criado anteriormente, e em seguida clicar em *Pin.*

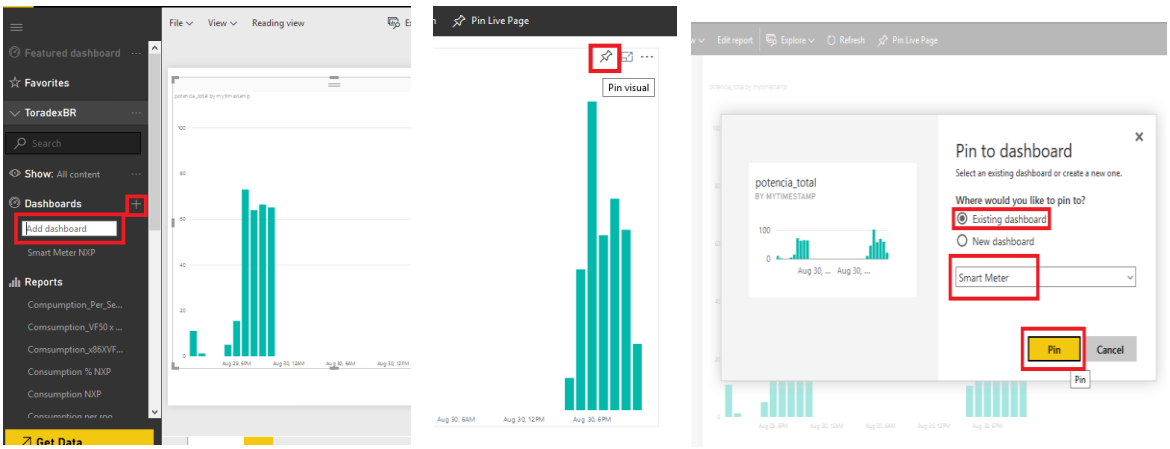


Figura 24: a)Criar Dashboard; b)Opção para fixar o gráfico criado no Dashboard; c)Configuração para fixar o gráfico no Dashboard[53]

Desta maneira os Dashboard criado serão atualizados automaticamente a cada vez que o *Stream Analytics* receber os dados provenientes do IoT Hub e enviá-los para o Power BI.

4- Resultados e Discussões

Nesta seção serão apresentados os resultados obtidos neste projeto, bem como a discussão dos mesmos.

4.1 – Resultados

A utilização do módulo Colibri VF50 teve desempenho satisfatório ao adquirir os dados em intervalos de 3 segundos e posteriormente enviá-los para a nuvem, utilizando as *APIs* do serviço do *Azure* para tal finalidade.

Após recebimento da mensagem pelo IoT Hub, a mesma é enviada ao *Stream Analytics* , o qual realiza o processamento destes dados e encaminha-os para o Power BI, onde os mesmos são exibidos através de gráficos. Este processo apresentou resultados satisfatórios, porém foi observado uma pequena latência entre o recebimento da mensagem pelo IoT Hub e a exibição da mesma através do Power BI. Esta latência foi observada quando, ao acender uma lâmpada, conectada ao quadro de tomadas, o gráfico do tipo Gauge, que mede corrente instantânea, demorou alguns segundos para atualização, como não houve atraso no envio da mensagem para nuvem, este atraso ocorreu no período que compreende o recebimento da mensagem pelo IoT Hub até a exibição do mesmo através do Power BI, porém o mesmo não pode ser identificado com precisão.

A ferramenta Power BI possibilitou uma exibição mais clara e eficiente dos dados enquistados através de diferentes tipos de gráficos. Abaixo estão descritos cada tipo de gráfico obtidos através da leitura destes dados.

1) Gráfico de linhas que exibe o consumo de cada sensor ao longo do tempo, o período de tempo escolhido foi do dia 4 ao dia 25 de setembro.

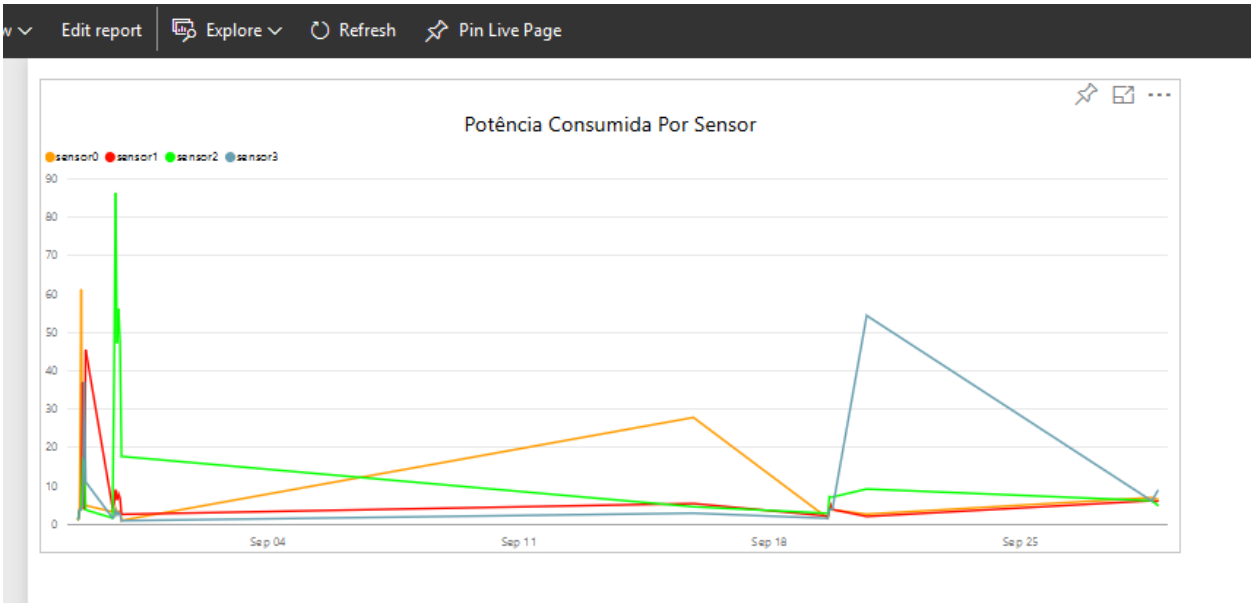


Figura 25: Gráfico de consumo por sensor[54]

2) O gráfico de barras exibe a soma do consumo de todos os sensores por hora, ou seja consumo total por hora, o período de tempo escolhido foi do dia 29 ao dia 31 de agosto.

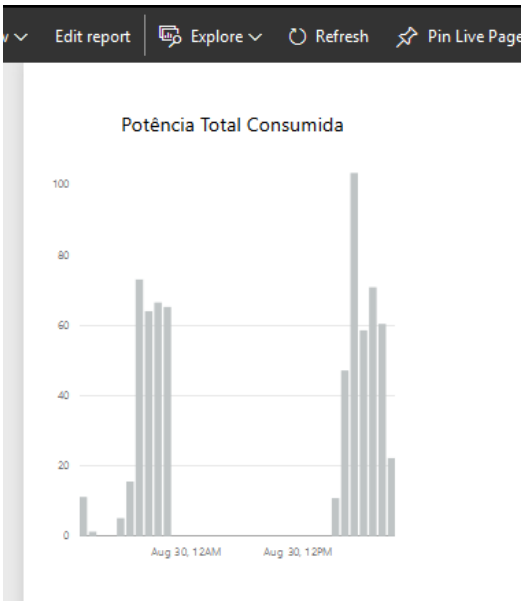


Figura 26: Gráfico de consumo total[55]

3) O gráfico treemap, que exibi dados hierárquicos através de retângulos aninhados, exibe o consumo por cômodo da casa. Isto foi possível através da *Query* do *Stream Analytics* , onde

cada sensor foi denominado como sendo instalado em um cômodo da casa, ilustrando assim uma aplicação onde cada sensor é responsável por medir o consumo de cada cômodo.

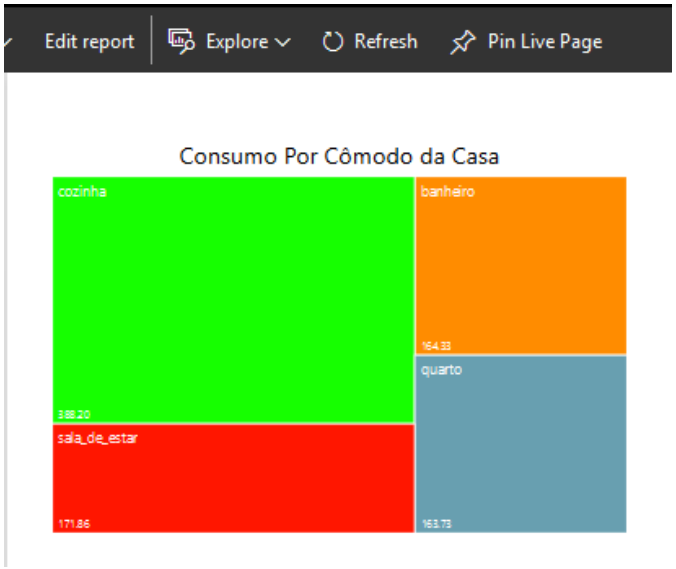


Figura 27: Gráfico de consumo por cômodo da casa[56]

4) Como o projeto consiste na medição de um quadro de tomadas tem-se a possibilidade de ligar tanto o módulo VF50 quanto o notebook, e a partir destes dados medir o consumo comparativo de ambas as tecnologias utilizadas.

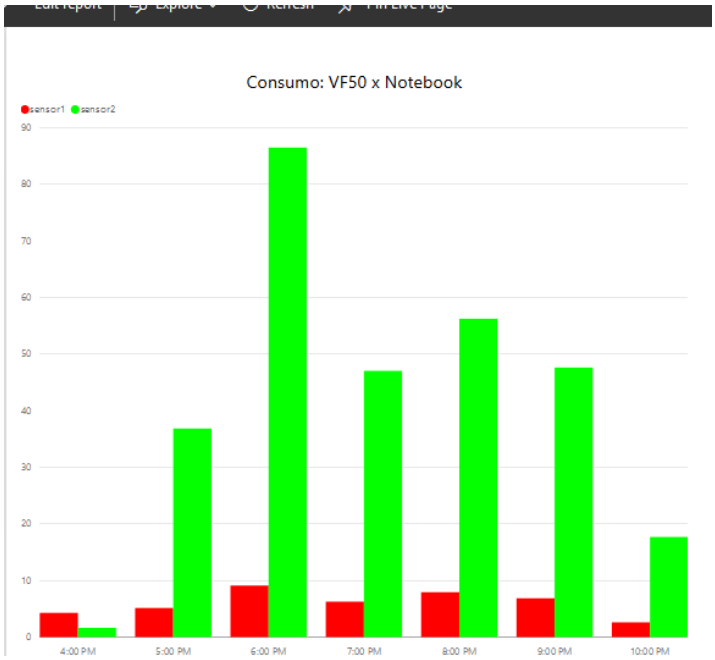


Figura 28: Gráfico comparativo entre o consumo do computador e do módulo VF50[57]

5) O gráfico de setores exibe o consumo comparativo de cada sensor, o período de tempo medido foi do dia 28 ao dia 31 de agosto.

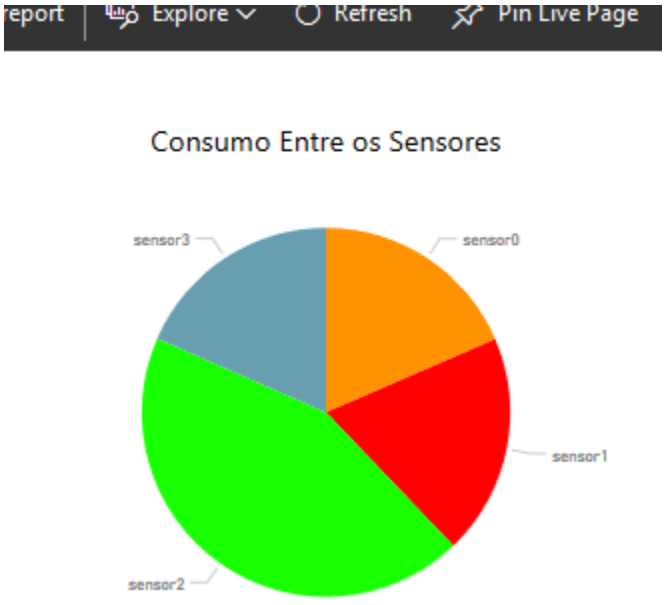
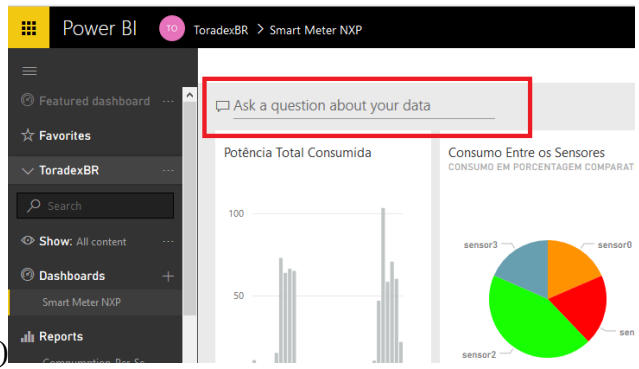


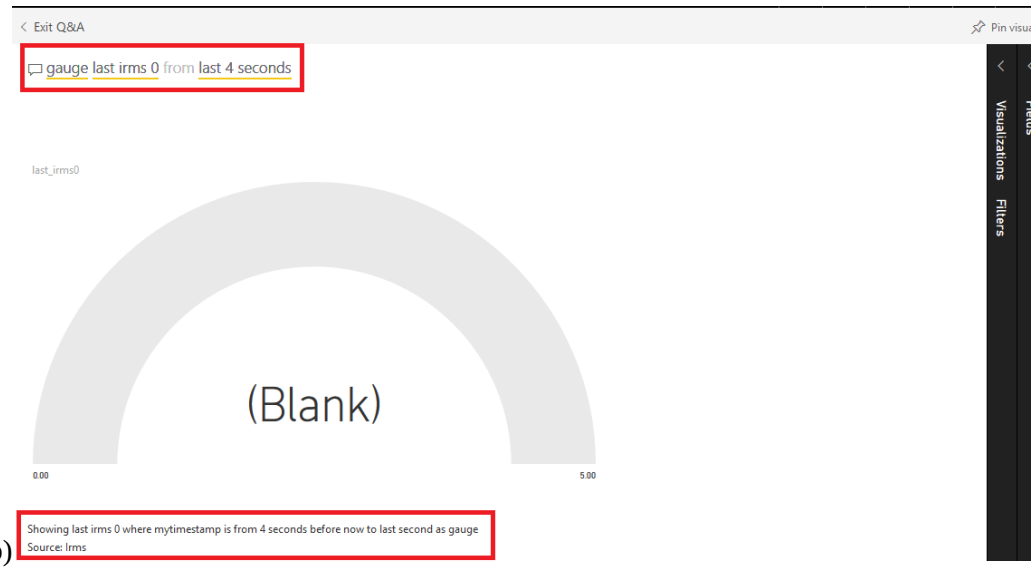
Figura 29: Gráfico de consumo entre os sensores[58]

6) Este último gráfico do tipo Gauge que mede a corrente Irms instantânea de cada sensor foi obtida através da função *Ask a question about your data*, presente no próprio Dashboard, a qual permite a elaboração de uma questão sobre os dados presentes no mesmo. Ao criar uma questão a resposta gerada pode ser uma informação ou gráfico, que poderão ser úteis em sua análise. Para obtenção deste gráfico foi utilizada a pergunta, **gauge last irms 0 from last 3 seconds**, e o resultado foi um gráfico do tipo Gauge que atualizava a corrente a cada envio do sensor.

a)



b)



c)

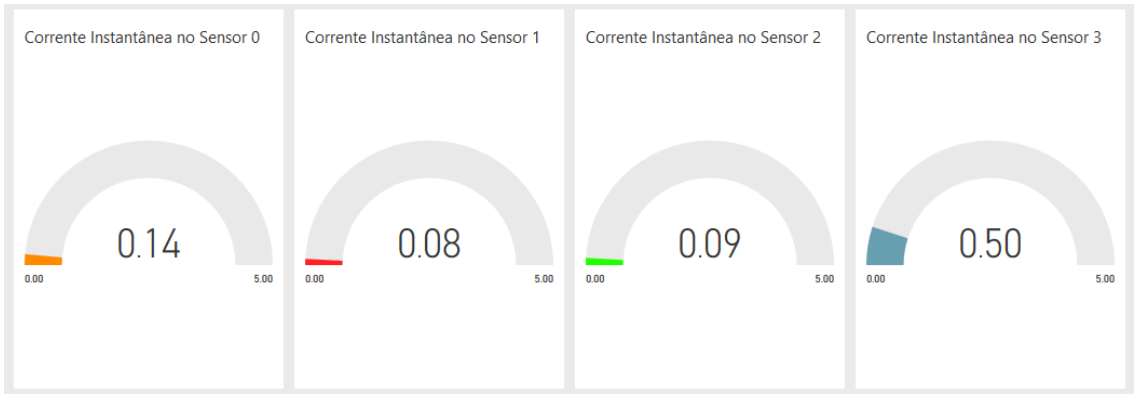
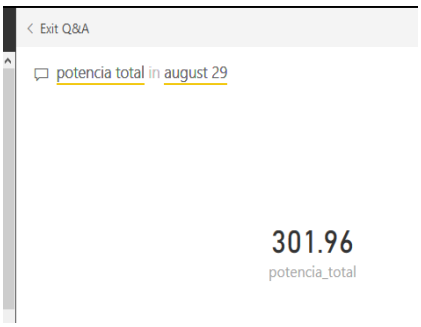


Figura 30: a) Campo *Ask a question about your data* b) Gráfico obtido através da pergunta *gauge last irms 0 from last 3 seconds* c) Gráficos de consumo instantâneo de corrente em cada sensor[59]

Esta ferramenta de pergunta, como a utilizada para gerar o gráfico do tipo Gauge descrita acima, pode gerar perguntas genéricas sobre a potência total consumida e a média total da potência consumida em agosto, e até mesmo perguntas mais específicas como a média total de cada sensor por cômodo consumida no dia 29 de agosto, dentre outras possíveis perguntas, como pode-se observar nos gráficos abaixo. Ferramenta esta que constitui em uma importante forma de análise em tempo real de um dado específico de maneira instantânea.



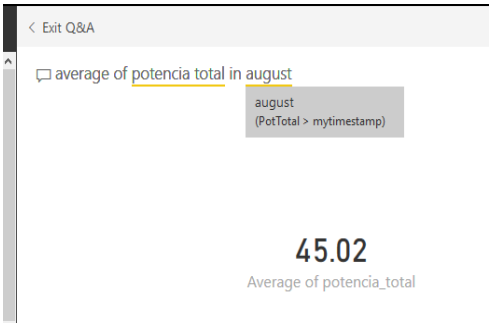


Figura 31: a) Dado da potência total consumida em agosto b) Dado de média da potência total consumida em agosto c) Dado de média total de cada sensor por cômodo consumida no dia 29 de agosto[60]

Através dos gráficos descritos acima somados a recursos de textos e figuras, foi obtido o Dashboard final como ilustrado na Figura 32. O qual exibe de maneira gráfica e de fácil entendimento os gastos de energia por sensor e o consumo total dos mesmos ao longo do tempo, munindo assim o consumidor de uma ferramenta em tempo real para a verificação do consumo de energia. Na Figura 33 é apresentado o protótipo final utilizado para a aquisição de dados.

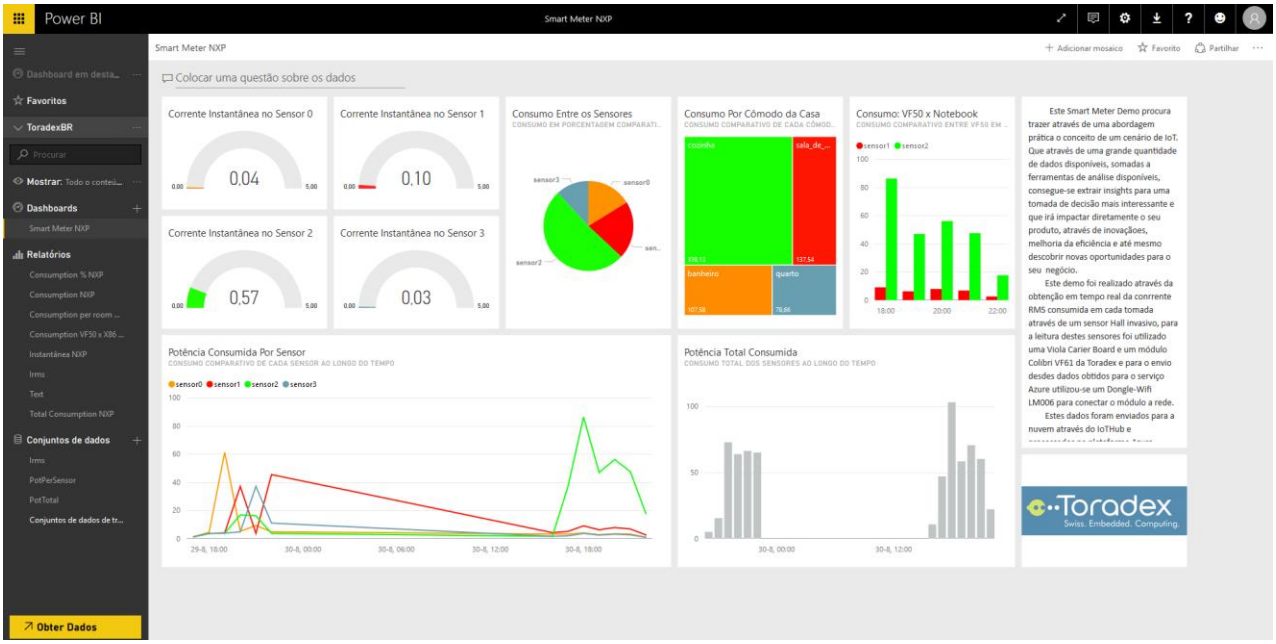


Figura 32: Dashboard final do projeto[61]

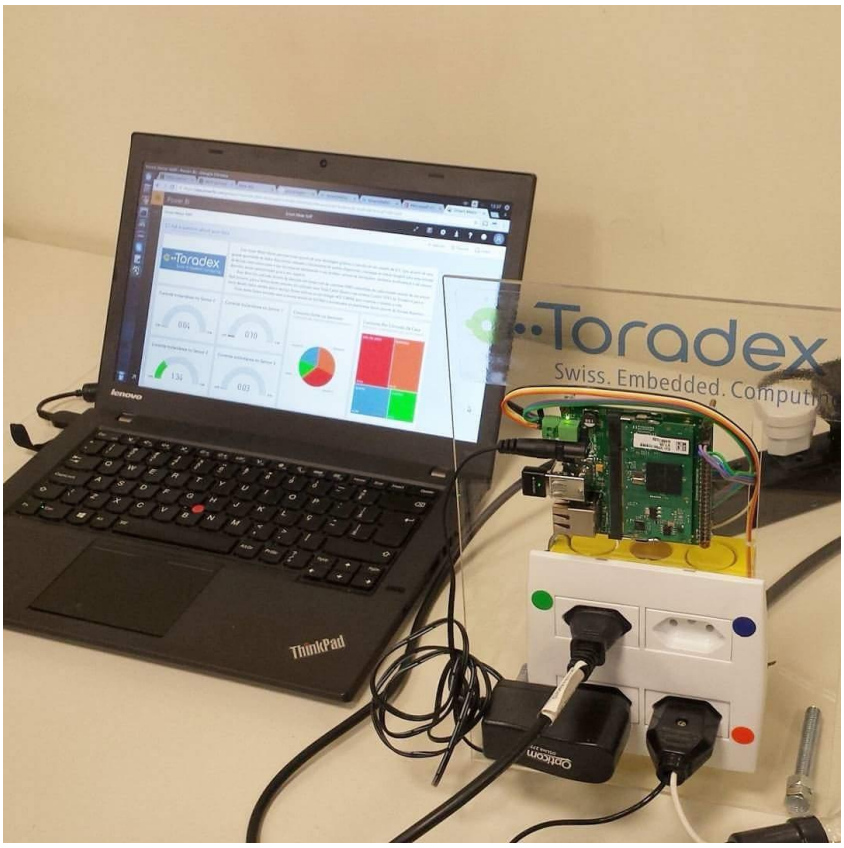


Figura 33: Protótipo final Smart Meter [38]

4.2 Discussão

A utilização do SoM Colibri VF50 juntamente com placa base Viola Plus obtiveram um processamento satisfatório, pois sendo o SoM VF50, dentre a família Colibri, um dos módulos com a menor capacidade de processamento, foi possível instalar todas as SDKs do IoT Hub, onde o programa que utiliza estes recursos para enviar as mensagens para o *Azure*, funcionou de maneira ininterrupta e sem travamentos durante sua execução, levando-se em consideração que o mesmo realiza a leitura dos sensores e envia os dados de 4 em 4 segundos para a nuvem. Porém se o produto necessitar de um processamento superior, a solução poderá facilmente ser substituída por um módulo que possui processamento superior, tais como Colibri IMX6 ou Colibri T30. O que possibilita a implementação de uma solução mais complexa sem a necessidade de replanejamento do hardware, devido a compatibilidade de pinos entre módulos da família Colibri.

A utilização da plataforma *Azure* apresentou resultados condizentes com o esperado, uma vez que este trabalho pode ser implementado com êxito, em um ambiente que possuía uma constante conectividade com a internet, onde tal plataforma dispunha de ferramentas diversas para implementação em diferentes cenários e tratamento eficiente dos dados recebidos. O serviço Power BI apresentou grande versatilidade na elaboração de diferentes gráficos, apresentando ferramentas de grande utilidade na extração instantânea de informações relevantes para análise e de grande auxilio na tomada de decisões utilizando diversas fontes e tipos de dados.

Inserindo neste cenário soluções, tais como SoMs e soluções em nuvem, contribuem no âmbito de reduzir o tempo de TtM de um produto. Onde tais soluções reduzem este tempo na área de desenvolvimento de hardware, devido ao fato de tais sistemas já possuirem prontas as etapas mais complexas do projeto, tais como a parte de memória e roteamento de sinais do microprocessador, caracterizando-se assim uma ferramenta que pode ser utilizada para reduzir este tempo de desenvolvimento da placa, quando este desenvolvimento for realizado em uma solução onde não se possui um conhecimento prévio na área, e partir para uma solução onde estão disponíveis todas as saídas de sinais prontas para a utilização, sendo necessário somente o projeto da placa base ou a utilização de uma disponibilizada pelo próprio fabricante de SoM.

Soluções em nuvem também reduzem este tempo de TtM de um produto, devido ao fato de as mesmas serem utilizadas e consequentemente cobradas conforme a necessidade de uso, otimizando assim os gastos com recursos sobressalentes ao projeto.

Segundo a literatura *feedbacks* sobre o consumo de energia de uma residência consiste em uma importante ferramenta para redução do consumo de energia elétrica [69][70][3][5][71], podendo ser exibida de diferentes maneiras, tais como: unidade monetária da energia consumida[69], diferentes unidades de

tempo, tais como mensal e anual[71]. Tais *feedbacks* deverão ser exibidos de maneira simples e visível para o consumidor, munindo-o de uma ferramenta que possa comparar o consumo de sua casa e que envolva todos os moradores no âmbito de economizar energia[5]. Neste ambito a ferramenta Power BI apresentou resultados satisfatórios, umas vez que a mesma dispões de diversas ferramentas para elaboração de diferentes gráficos e que apresentam de forma clara e objetiva o consumo total de energia, o consumo por sensor, em uma situação hipotética o consumo por cômodos de uma residência, bem como ferramentas para análise imediata de dados, tais como *Ask Question about your data*, possibilitando a atualização instantânea dos dados de consumo de uma casa.

Porém como apresentado em outros trabalhos tal ferramenta torna-se mais efetiva ao aliar-se outras medidas no âmbito de economia de energia, tais como campanhas midiáticas, conscientização das ações de economia de energia no ambito socio-cultural[3].

5 – Conclusão

Este projeto teve como objetivo unir o conhecimento adquirido durante a graduação e o período de estágio, através de um projeto que utiliza as ferramentas disponíveis no mercado e que auxiliam na redução do TtM de um produto, viabilizando a concepção do mesmo em um tempo mais reduzido se comparado ao desenvolvimento completo da solução.

Utilizando um SoM Colibri VF50 + paca base Viola Plus juntamente com plataforma *Azure* e o serviço Power BI foi concebido um projeto de um Smart Meter. Neste projeto foram adquiridos dados de corrente de um quadro de tomadas, tais dados enviados para a nuvem e sendo utilizando os serviços da *Microsoft* foram criados gráficos em um *dashboard*, os quais exibem dados de consumo instantâneo de energia de uma residência.

Através da utilização de tais ferramentas obteve-se êxito na execução do trabalho, uma vez que estes *dashboards* foram obtidos com o intuito de munir o cliente de uma ferramenta na qual pode ser analisado o consumo de uma casa em diferentes cômodos bem como a potência total consumida instantaneamente. Em trabalhos futuros deverão ser realizados estudos que visem identificar uma forma mais otimizada para exibição dos dados para o consumidor. Exibindo os gráficos de maneira clara e objetiva conforme a realidade de consumo do consumidor brasileiro e como abordado em outros trabalhos, aliar informação de consumo a dicas de como economizar energia para usuários iniciais[70][69]. Tal análise não foi abordada neste trabalho, devido ao fato de o objetivo deste consistir na a apresentação uma ferramenta, que neste trabalho foi aplicada por meio da concepção de um Smart Meter, porém este trabalho poderá ser reproduzido e servir como base para outras aplicações que visem o cenário de IoT.

Inicialmente este projeto foi concebido visando a aplicação no cenário residencial, desta maneira este projeto poderia ser implementado para outras plataformas mais acessíveis como Raspberry Pi sem muitas modificações. Porém este projeto também visa a implementação de uma ferramenta que também visa uma aplicação voltada para a indústria e que necessita de características tais como: produtos disponíveis com ciclo de vida de no mínimo 10 anos, estoque independente em múltiplas localidades geográficas, padrões que robustos e que suportam diversas interfaces, padrões de qualidade como teste final de linha, testes de choque e vibração [65], dentre outras vantagens. Uma vez que tais soluções podem ser utilizadas em cenários industriais.

Tal solução também contribuem no âmbito de reduzir os riscos da implementação de um novo produto, devido ao fato de abstrair grande parte do desenvolvimento, o mesmo poderá ser lançado rapidamente no mercado e com reduzidos gastos com recursos de tempo e financeiros na etapa de desenvolvimento. Características estas citadas em outros trabalhos e que caracterizam como vantagens ao ser utilizada em produtos, reduzindo gastos com recursos sobressalentes de projetos [14][67][68].

Através deste Smart Meter abordou-se também o conceito de IoT, embora esteja tão popular atualmente, o mesmo vinha sendo aplicado há algum tempo. Porém o grande potencial que as empresas encontram em tal cenário é a capacidade de obter a maior quantidade de dados possíveis, onde tais dados servirão para análise e posterior extração de ideias e decisões mais bem fundamentadas e que irão auxiliar no dia-a-dia das empresas bem como na otimização e/ou automatização de processos.

Visando tal cenário este trabalho abordou-se o conceito de IoT de maneira completa, desde a aquisição dos dados, envio para a nuvem, tratamento dos dados e posterior exibição dos mesmos para o consumidor de maneira clara e objetiva, munindo-o de uma ferramenta que o ajude a otimizar sua tomada de decisão, que neste projeto consiste na economia de energia de uma residência.

Outras possíveis futuras otimizações deste projeto, poderão ser obtidas através de ferramentas tais como Machine Learning do *Azure*, onde além de dados de consumo tem-se a possibilidade de aliar outros dados, como por exemplo de temperatura e até mesmo de consumo de água, e fazer um sistema que preveja o consumo de energia de uma casa em diferentes áreas e levando em consideração diferentes fatores.

6 – Referências

[1] DINCER, Ibrahim. Renewable energy and sustainable development: a crucial review. **Renewable and Sustainable Energy Reviews**, v. 4, n. 2, p. 157-175, jun. 2000.

[2] OMER, Abdeen Mustafa. Energy, environment and sustainable development. **Renewable and Sustainable Energy Reviews**, v. 12, n. 9, p. 2265-2300, dez. 2008.

[3] STEG, Linda. Promoting household energy conservation. **Energy Policy**, v. 36, p. 4449-4453, out. 2008.

[4] EDSON, Barb; Creating the internet of your things.
Disponível em: <https://www.google.com.br/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&ved=0ahUKEwjssdXUtITQAhWJHJAKHcWqClUQFgghMAA&url=http%3A%2F%2Fdownload.*Microsoft*.com%2Fdownload%2FE%2F1%2FF%2FE1FFDADF-C0FF-4E72-A834-B173A079F393%2F*Microsoft*_Internet_of_Things_White_Paper.pdf&usg=AFQjCNHoFpL5a4lqM90p8f7Plh8bIfVE5A&sig2=l5kjbDJVa0SeIsKXVatHoQ>
Acesso em 4 de setembro de 2016

[5] HARGREAVES, Tom; NYE, Michael; BURGESS, Jacquelin. Making energy visible: A qualitative field study of howhouseholders interact with feedback from smart energy monitors. **Energy Policy**, v. 38, p. 6111-6119, jul. 2010.

[6] MIORANDI, D. et al. Internet of things: Vision, applications and research challenges. **Ad Hoc Networks**, v. 10, p. 1497-1516, abr. 2012.

[7] VESEY, Joseph T; Time-to-Market: Put Speed in Product Development. **Industrial marketing management**, v. 21, p. 151, Maio, 1992.

[8] SILVA, Niágara; FÉRES, José; LÍRIO, Viviane; Análise da Estrutura da Demanda de Energia Elétrica Residencial Segundo os Quantis de Consumo
Disponível em < http://www.ipea.gov.br/agencia/images/stories/PDFs/radar/121114_radar22_cap6 >
Acessado em 29 de outubro de 2016

[9] BARR, Michael ; MASSA, Anthony. **Programming Embedded Systems**: with C and GNU Development Tools. 1 ed. [S.L.]: O'Reilly, 1999. 9-11 p.

[10] NOERGAARD, Tammy. **Embedded Systems Architecture**: A Comprehensive Guide for Engineers and Programmers. Burlington, MA, USA: Elsevier, 2015. 1 p.

[11] Colibri VF50 < http://docs.toradex.com/101355-colibri-vf50-datasheet.pdf>

[12] HALLINAN, Christopher. **Embedded Linux Primer**: a practical real-word approach. 2 ed. Boston: prentice hall, 2011. 137 p.

[13] RAGHAVAN, P.; LAD, Amol; NEELAKANDAN, Sriram. **Embedded Linux System Design and Development**. 1 ed., Auerbach Publications, 2005. 1-2 p.

[14] MARSTON, S. et al. Cloud computing — The business perspective. **Decision Support Systems**, [S.L], v. 51, p. 176-189, dez. 2010.

[15] BOSWARTHICK, David; ELLOUMI, Omar; HERSENT, Olivier. **M2M Communications**: A Systems Approach. 1 ed. [S.L.]: Wiley, 2012. 1-5 p.

[16] Forbes < http://www.forbes.com/sites/jacobmorgan/2014/05/13/simple-explanation-internet-things-that-anyone-can-understand/#67d9bba16828 >

[17] *Azure*< https://azure.*Microsoft* .com/en-us/ >

[18] IoT Hub < https://azure.*Microsoft* .com/pt-br/documentation/articles/iot-hub-what-is-iot-hub/ >

[19] *Stream Analytics* < https://azure.*Microsoft* .com/pt-br/documentation/articles/*Stream -Analytics* - introduction/ >

[20] *Stream Analytics Query* Language Reference
< https://msdn.*Microsoft* .com/en-us/library/azure/dn834998.aspx >

[21] Power BI < https://powerbi.*Microsoft* .com/pt-br/what-is-power-bi/ >

[22] Allegro Hall-Effect Sensor ICs < http://www.allegromicro.com/ja-JP/Design-Center/Technical-Documents/Hall-Effect-Sensor-IC-Publications/Allegro-Hall-Effect-Sensor-ICs.aspx >

[23] Allego ACS712 < http://www.allegromicro.com/~/media/files/datasheets/acs712-datasheet.ashx >

[24] Placa base Viola Plus < http://docs.toradex.com/102879-colibri-arm-viola-carrier-board-datasheet.pdf >

[25] Node JS < https://pt.wikipedia.org/wiki/Node.js >

[26] TEIXEIRA, Pedro. **Professional Node.js**: Building Javascript Based Scalable Software. 1 ed.: Wrox, 2012. 1-2 p.

[27] Nodemon < https://udgwebdev.com/nodejs-ou-nodemon >

[28] Wi-Fi LM006 < http://docs.toradex.com/102747-lm006-usb-wifi-datasheet.pdf >

[29] Toradex Linux Image < http://developer1.toradex.com/files/toradex-dev/uploads/media/Colibri/Linux/Images/ >

[30] Flashing Embedded Linux to Vybrid Modules < http://developer.toradex.com/knowledge-base/flashing-linux-on-vybrid-modules >

[31] ADC Linux- VF50 < http://developer.toradex.com/knowledge-base/adc-(linux)#Colibri_VFxx >

[32] Create your free *Azure*account < https://azure.*Microsoft* .com/en-us/free/ >

[33] Create Internet of things < https://portal.azure.com/?whr=live.com#create/hub >

[34] Introdução ao Hub IoT do *Azure*para Node.js < https://azure.*Microsoft* .com/pt-br/documentation/articles/iot-hub-node-node-getstarted/>

[35] Visão geral do *Azure*Resource Manager
< https://azure.*Microsoft* .com/pt-br/documentation/articles/resource-group-overview/ >

[36]Create a IoT Hub < https://portal.azure.com/?whr=live.com#create/*Microsoft* .IotHub >

[37] Página inicial IoT Hub < https://portal.azure.com/?whr=live.com#resource/subscriptions/781e30d8-def1-445c-910d-bb8ac68344a0/resourceGroups/toradex_rs/providers/*Microsoft* .*Stream Analytics* /*Stream* ingjobs/SmartMeter >

[38] Fonte : Próprio autor

[39] Device Explorer < https://portal.azure.com/?whr=live.com#resource/subscriptions/781e30d8-def1-445c-910d-bb8ac68344a0/resourcegroups/toradex_rs/providers/*Microsoft* .Devices/IotHubs/SmartMeter/Overview >

[40] Enviando dados para a nuvem com *Azure*IoT Hub < http://www.embarcados.com.br/dados-na-nuvem-com-azure-iot-hub/ <

[41] Power BI < https://portal.office.com/signup?sku=a403ebcc-fae0-4ca2-8c8c-7a907fd6c235&email&ru=https%3A%2F%2Fapp.powerbi.com%3Fpbi_source%3Dweb%26redirectedFromSignup%3D1%26noSignUpCheck%3D1%26pbi_source_id%3D41a1f6735f8e1748bec132de02ba072 >
[42]Cadastro Power BI < https://portal.office.com/signup?sku=a403ebcc-fae0-4ca2-8c8c-7a907fd6c235&email&ru=https%3A%2F%2Fapp.powerbi.com%3Fpbi_source%3Dweb%26redirectedFromSignup%3D1%26noSignUpCheck%3D1%26pbi_source_id%3D41a1f6735f8e1748bec132de02ba072c >

[43] Cadastro Power BI < https://portal.office.com/signup?sku=a403ebcc-fae0-4ca2-8c8c-7a907fd6c235&email&ru=https%3A%2F%2Fapp.powerbi.com%3Fpbi_source%3Dweb%26redirectedFromSignup%3D1%26noSignUpCheck%3D1%26pbi_source_id%3D41a1f6735f8e1748bec132de02ba072c >

[44] Página inicial Powe BI < https://app.powerbi.com/groups/me/dashboards/9581358c-c648-449a-b2e7-5086e407cfe9>

[45] Página *Stream Analytics* Job < https://portal.azure.com/?whr=live.com#create/*Microsoft .Stream Analytics* Job >

[46] Adicionar Entrada *Stream Analytics* < https://portal.azure.com/?whr=live.com#resource/subscriptions/781e30d8-def1-445c-910d-bb8ac68344a0/resourceGroups/toradex_rs/providers/*Microsoft .Stream Analytics* /*Stream* ingjobs/SmartMeter/overview >

[47] Adicionar Saída *Stream Analytics* < https://portal.azure.com/?whr=live.com#resource/subscriptions/781e30d8-def1-445c-910d-bb8ac68344a0/resourceGroups/toradex_rs/providers/*Microsoft .Stream Analytics* /*Stream* ingjobs/SmartMeter/overview>

[48] *Query Stream Analytics* < https://portal.azure.com/?whr=live.com#resource/subscriptions/781e30d8-def1-445c-910d-bb8ac68344a0/resourceGroups/toradex_rs/providers/*Microsoft .Stream Analytics* /*Stream* ingjobs/SmartMeter/overview>

[49] Página Inicial *Stream Analytics* < https://portal.azure.com/?whr=live.com#resource/subscriptions/781e30d8-def1-445c-910d-bb8ac68344a0/resourceGroups/toradex_rs/providers/*Microsoft .Stream Analytics* /*Stream* ingjobs/SmartMeter/overview >

[50] Documentação Power BI < https://powerbi.*Microsoft* .com/pt-br/documentation/powerbi-landing-page/ >

[51] Criar gráfico Power BI < https://app.powerbi.com/groups/1fa6b4d4-d991-4b30-babd-24938626989d/datasets/ba060613-2d6b-426f-8b66-cd745894a70d>

[52] Salvar gráfico em um Relatório < https://app.powerbi.com/groups/1fa6b4d4-d991-4b30-babd-24938626989d/datasets/ba060613-2d6b-426f-8b66-cd745894a70d >

[53] Salvar gráfico no Dashboard < https://app.powerbi.com/groups/1fa6b4d4-d991-4b30-babd-24938626989d/reports/50814ae9-cc69-4729-9e5d-61f0b3f1d571/ReportSection>

[54] Gráfico de linhas < https://app.powerbi.com/groups/1fa6b4d4-d991-4b30-babd-24938626989d/reports/b70d0a22-13d5-4c8b-b917-dbfb7e7e4be5/ReportSection>

[55] Gráfico de barras < https://app.powerbi.com/groups/1fa6b4d4-d991-4b30-babd-24938626989d/reports/902296fc-bc45-4d38-817c-5fe025ce3dac/ReportSection>

[56] Gráfico treemap < https://app.powerbi.com/groups/1fa6b4d4-d991-4b30-babd-24938626989d/reports/19dff487-0731-472f-957b-cbca0934dc96/ReportSection>

[57] Gráfico de barras < https://app.powerbi.com/groups/1fa6b4d4-d991-4b30-babd-24938626989d/reports/7b703146-93f5-4674-8705-3f71b8c7008e/ReportSection>

[58] Gráfico de setores < https://app.powerbi.com/groups/1fa6b4d4-d991-4b30-babd-24938626989d/reports/0f2438b4-aadf-447c-b07f-0bf17b1a4f06/ReportSection>

[59] Gráfico Gauge utilizando Perguntas < https://app.powerbi.com/groups/1fa6b4d4-d991-4b30-babd-24938626989d/dashboards/6e07bcda-0c95-45d9-88c8-05af77bb7ad4/qna?q=gauge%20last%20irms%203%20from%20last%204%20seconds%20>

[60] Dado obtido através de uma Pergunta < https://app.powerbi.com/groups/1fa6b4d4-d991-4b30-babd-24938626989d/dashboards/6e07bcda-0c95-45d9-88c8-05af77bb7ad4/qna?q=average%20of%20potencia%20total%20%20in%20august%20>

[61] AFONSO, P. et al. The influence of time-to-market and target costing in the new product development success. **International journal of production economics,** [S.L.], v. 115, p. 559-568, out. 2008.

[62] CARRILLO, Janice E.; FRANZA, Richard M.. Investing in product development and production capabilities: the crucial linkage between time-to-market and ramp-up time. **European journal of operational research,** [S.L.], v. 171, p. 536-556, jun. 2006.

[63] ANTHONY, Richard. **Systems Programming**: Designing and Developing Distributed Applications. 1 ed. Waltham, MA, USA: Morgan Kaufmann, 2015. 465 p.

[64] HTTP < https://www.jmarshall.com/easy/http/#whatis >

[65] Toradex < https://www.toradex.com/pt_br/how-to-choose-system-computer-on-module-partner>

[66] MILLSON, Murray R.; RAJ, S.P.; WILEMON, David. A survey of major approaches for accelerating new product development. **Journal of Product Innovation Management**, [S.L], v. 9, n. 1, p. 53-69, mai. 2002.

[67] ZHANG, Qi; CHENG, Lu; BOUTABA, Raouf. Cloud computing: state-of-the-art and research challenges. **Journal of Internet Services and Applications**, [S.L], v. 1, n. 1, p. 1-18, abr. 2010.

[68] O., Omoniyi Temitope.. Cloud computing for business. **Msc computing & management,** 2009-2010.

[69] ABRAHAMSE, W. et al. A review of intervention studies aimed at household energy conservation. **Journal of Environmental Psychology**, [S.L], v. 25, n. 3, p. 273-291, nov. 2005.

[70] WOOD, G.; NEWBOROUGH, M.. Dynamic energy-consumption indicators for domestic appliances: environment, behaviour and design. **Energy and Buildings**, [S.L], v. 35, n. 8, p. 821-841, jan. 2003.

[71] WOOD, G.; NEWBOROUGH, M.. Energy-use information transfer for intelligent homes: Enabling energy conservation with central and local displays. **Energy and Buildings**, [S.L], v. 39, n. 4, p. 495-503, out.

7 – Apêndice

1- Algoritmo que envia os dados para a nuvem SendData

```
'use strict';
var Protocol = require('azure-iot-device-http').Http;   // Protocolo de comunicação
var Client = require('azure-iot-device').Client;
var Message = require('azure-iot-device').Message;
var fs = require('fs');
var                 ADC0 = '0',
                    ADC1 = '0',
                    ADC2 = '1',
                    ADC3 = '0',
                    VoltsPorUnidade= 0.185,
                    i,
                    V_aux = require('vectors/mag')(60),
                    V_aux,
                    minValue,
                    maxValue,
                    Vp,
                    Sqrt,
                    Vrms,
                    Arms,
                    ArmsS0,
                    ArmsS1,
                    ArmsS2,
                    ArmsS3,
```

```
                    PS0,
                    PS1,
                    PS2,
                    PS3;
var connectionString = "HostName=yourHostName.azure-devices.net;DeviceId=yourDevice;SharedAccessKey=yourSharedAccessKey";  //Deverá ser substituída pela obtida no IoT Hub Devices
var client = Client.fromConnectionString(connectionString, Protocol);
var sendInterval = {timerGet: 3000, timerSend: 3000};//loop handler
var timenow,Armss0,Armss1,Armss2,Armss3,Ps0,Ps1,Ps2,Ps3 ;

// Recursividade que chama as funções para leitura e envio dos sensores para a nuvem
sendInterval.handlerGet = setInterval(getAllSensors, sendInterval.timerGet);
sendInterval.handlerSend = setInterval(sendToIotHub, sendInterval.timerSend);
//Função para leitura dos sensores
function getAllSensors() {
        Vrmsl = 125;
        var d = new Date();
        timenow = d.getTime();// pegar a data do módulo embarcado
        //console.log("Inicia Leitura...")
        ArmsS0 = rdADC(ADC0,'8');
        PS0=ArmsS0*Vrmsl;
        ArmsS1 = rdADC(ADC1,'8');
        PS1=ArmsS1*Vrmsl;
        ArmsS2 = rdADC(ADC2,'9');
        PS2=ArmsS2*Vrmsl;
        ArmsS3 = rdADC(ADC3,'9');
        PS3=ArmsS3*Vrmsl;
}
```

```
//Ler dado dos sensors por meio do canal ADC
function rdADC(number){
	/*---------- Inicio Leitura AC Sensor ----------*/
	maxValue=0;
	minValue=4096;
	for(i=0;i<100;i++){

	V_aux[i]=fs.readFileSync('/sys/bus/iio/devices/iio:device'+pin+'/in_voltage'+number+'_raw');
		if (V_aux[i]<minValue){
			minValue=V_aux[i];
		}
		if(V_aux[i]>maxValue){
			maxValue=V_aux[i];
		}
	}
	Vp=((maxValue-minValue)*3.3)/4096
	Sqrt= math.sqrt(2);
	Vrms=Vp/(2* Sqrt);
	Arms=(Vrms)/VoltsPorUnidade;
	return Arms;
}
function sendToIotHub() {
	// encapsular a mensagem no formato JSON
	var data = JSON.stringify({
		ObjectName: 'toradexMeter',
		ObjectType: 'SensorTagEvent',
		Armss0: ArmsS0,
		Ps0: PS0,
```

```
                    Armss1: ArmsS1,
                    Ps1: PS1,
                    Armss2: ArmsS2,
                    Ps2: PS2,
                    Armss3: ArmsS3,
                    Ps3: PS3,
                    boardTime: timenow
            });
            var message = new Message(data);// Encapsula a mensagem a ser enviada
            message.properties.add('myproperty', 'myvalue');
            console.log('sending message to the IoT Hub: ');// Feedback da mensagem enviada
            console.log(data);
            client.sendEvent(message, printResultFor('send'));// Envia a mensagem ao IoT Hub
}

//Função auxiliar que imprime os resultados no console do módulo
function printResultFor(op) {
 return function printResult(err, body, res) {
  if (err) console.log(op + ' error: ' + err.toString());
  if (res){
   console.log(op + ' response: ' + res);
   console.log(body);
  }
 };
}
```

Algoritmo 1: Envia os dados para a nuvem SendData[40]

2- Programa utilizado na *Query* do *Stream Analytics*

```
-- Irms
SELECT
System.Timestamp AS myTimestamp,
  ObjectName as SensorSmartMeter,

  --Corrente ao longo do tempo
  Armss0 AS Sensor0,
  Armss1 AS Sensor1,
  Armss2 AS Sensor2,
  Armss3 AS Sensor3

INTO
  DataSmartMeter
FROM
  smartmeter
ObjectName


-- Consumo [KWh]
SELECT
System.Timestamp AS myTimestamp,
  ObjectName as SensorSmartMeter2,

  -- Consumo total por sensor ao longo do tempo
  SUM(Ps0)/1000 AS Sensor0,
  SUM(Ps1)/1000 AS Sensor1,
  Sum(Ps2)/1000 AS Sensor2,
  SUM(Ps3)/1000 AS Sensor3,


  -- Consumo total por cômodos da casa
  SUM(Ps0)/1000 AS Banheiro,
  SUM(Ps1)/1000 AS Sala_De_Estar,
  Sum(Ps2)/1000 AS Cozinha,
  SUM(Ps3)/1000 AS Quarto
```

```
INTO
  DataSmartMeter2
FROM
  smartmeter
GROUP BY
  TumblingWindow(hh, 1),
ObjectName


-- Consumo [KWh]
SELECT
System.Timestamp AS myTimestamp,
  ObjectName as SensorSmartMeter3,

  --Consumo total de todos os sensores ao longo do tempo
  ((SUM(Ps0)/1000)+(SUM(Ps1)/1000)+(SUM(Ps2)/1000)+(SUM(Ps3)/1000)) AS
Potencia_Total

INTO
  DataSmartMeter3
FROM
  smartmeter
GROUP BY
  TumblingWindow(hh, 1),
ObjectName
```

Algoritmo 2: *Query Stream Analytics* [38]