

**UNIVERSIDADE ESTADUAL PAULISTA "JÚLIO DE MESQUITA FILHO"
FACULDADE DE ARQUITETURA, ARTES E COMUNICAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO EM MÍDIA E TECNOLOGIA**

JOSÉ EUGÊNIO DE MIRA

**DESENVOLVIMENTO DE UM MODELO DE NEURÔNIO ARTIFICIAL COM
MICROCONTROLADOR WI-FI AUTÔNOMO COM COMUNICAÇÃO M2M PARA
IOT**

Bauru

2019

JOSÉ EUGÊNIO DE MIRA

**DESENVOLVIMENTO DE UM MODELO DE NEURÔNIO ARTIFICIAL COM
MICROCONTROLADOR WI-FI AUTÔNOMO COM COMUNICAÇÃO M2M PARA
IOT**

Trabalho de Conclusão de Mestrado apresentado ao Programa de Pós-graduação em Mídia e Tecnologia, da Faculdade de Arquitetura, Artes e Comunicação – FAAC, Universidade Estadual Paulista “Júlio de Mesquita Filho” – UNESP, para obtenção do título de Mestre em Mídia e Tecnologia sob a orientação do Prof. Titular João Fernando Marar.

Bauru

2019

Mira, José Eugênio.

DESENVOLVIMENTO DE UM MODELO DE NEURÔNIO ARTIFICIAL COM MICROCONTROLADOR WI-FI AUTÔNOMO COM COMUNICAÇÃO M2M PARA IOT / José Eugênio de Mira, 2019. 71 f.: il.

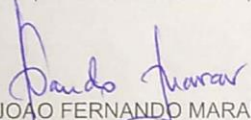
Orientador: João Fernando Marar.

Dissertação (Mestrado) - Universidade Estadual Paulista "Júlio de Mesquita Filho". Faculdade de Arquitetura, Artes e Comunicação, Bauru, 2019

1. Microcontrolador. 2. Rede Neural. 3. IoT.
I. Universidade Estadual Paulista "Júlio de Mesquita Filho". Faculdade de Arquitetura, Artes e Comunicação. II. Desenvolvimento de um modelo de neurônio artificial com microcontrolador Wi-Fi autônomo com comunicação M2M para IoT.

ATA DA DEFESA PÚBLICA DA DISSERTAÇÃO DE Mestrado DE JOSÉ EUGÊNIO DE MIRA, DISCENTE DO PROGRAMA DE PÓS-GRADUAÇÃO EM MÍDIA E TECNOLOGIA, DA FACULDADE DE ARQUITETURA, ARTES E COMUNICAÇÃO - CÂMPUS DE BAURU.

Aos 08 dias do mês de fevereiro do ano de 2019, às 14:00 horas, no(a) Auditório da Secretaria de Pós-Graduação da FAAC, reuniu-se a Comissão Examinadora da Defesa Pública, composta pelos seguintes membros: Prof. Dr. JOAO FERNANDO MARAR - Orientador(a) do(a) Departamento de Computação / Faculdade de Ciências - UNESP/Campus de Bauru, Prof^a. Dr^a. REGILENE APARECIDA SARZI RIBEIRO do(a) Departamento de Artes e Representação Gráfica / Faculdade de Arquitetura Artes e Comunicação de Bauru - Unesp, Prof. Dr. ANTONIO TADEU PELLISON do(a) Faculdade de Tecnologia de SP - FATEC - Campus de Bauru, sob a presidência do primeiro, a fim de proceder a arguição pública da DISSERTAÇÃO DE Mestrado de JOSÉ EUGÊNIO DE MIRA, intitulada **DESENVOLVIMENTO DE UM MODELO DE NEURÔNIO ARTIFICIAL COM MICROCONTROLADOR WI-FI AUTÔNOMO COM COMUNICAÇÃO M2M PARA IOT**. Após a exposição, o discente foi arguido oralmente pelos membros da Comissão Examinadora, tendo recebido o conceito final: Aprovado. Nada mais havendo, foi lavrada a presente ata, que após lida e aprovada, foi assinada pelos membros da Comissão Examinadora.


Prof. Dr. JOAO FERNANDO MARAR


Prof. Dr^a. REGILENE APARECIDA SARZI RIBEIRO

Prof. Dr. ANTONIO TADEU PELLISON

Esse trabalho é dedicado a todos aqueles que têm um sonho e lutam por ele, acreditando ser possível. Dedico aos meus pais, Canuto e Luiza e aos meus irmãos Luciana e Canuto Junior e demais membros da minha família que acreditaram em meu potencial, me incentivaram e sempre insistiram para que eu continuasse meus estudos. Sem eles sequer a graduação seria possível, quiçá um Mestrado.

Dedico aos meus amigos de escola e de vida: Everton, Evandro, Cleber, Daniel, Rafael Henrique, Diogo, Arilson, Wellington e Rafael (*in memoriam*) e ao estigma nerd que me permitiu continuar longas leituras e aumentar minha paixão pelo conhecimento. Dedico aos meus colegas de trabalho da Fatec Bauru: Priscilla, Marcela, Paulo, Rafael, Ana Letícia, Cleiton, John, Ralf, Elaine, Torres assim como a outros professores, colegas e estagiários. Dedico ainda à Carol, que teve empatia com meus momentos de dedicação com esse projeto e empatia nos meus momentos de introspecção e preocupação.

Dedico aos colegas de Faculdade, aos colegas das empresas por onde passei, aos meus alunos e ex-alunos, pois o meu presente é fruto das experiências e aprendizados com essas pessoas no passado.

Por último, dedico ao Professor Doutor João Fernando Marar, que nos deixou órfãos de sua genialidade, porém amparados e inspirados pelo seu amor à ciência.

Que a força esteja com vocês.

AGRADECIMENTO

Agradeço primeiramente ao meu Orientador, Professor Titular Doutor João Fernando Marar, que em sua paixão pela inovação abraçou meu projeto e minha ideia de estudar alguns microcontroladores sem fio, sempre me apoiando em minhas loucuras e me advertindo para o domínio dos chineses nessa área. Em uma dessas infelizes contradições da vida, o Professor Marar tristemente nos deixou apenas dez dias após a defesa desse trabalho, porém seu legado acadêmico será levado adiante por nós, seus gratos orientados. Agradeço enormemente aos professores do programa de Pós-Graduação que tem estado conosco nessa caminhada, seja nos eventos ou nas rodas de conversa, mas principalmente àqueles que ministraram disciplinas: Professor Dr. Dino Magnoni e as Professoras Dr^a Regilene Sarzi e Dr^a Maria Cristina Gobbi, primeira pessoa com a qual troquei um e-mail sobre a participação no programa. Agradeço ainda à Unesp Bauru, à Faculdade de Arquitetura, Artes e Comunicação e aos funcionários da Seção Técnica de Pós-graduação, principalmente à Daniela, ao Helder e ao Sívio, todos sempre muito solícitos.

Não poderia deixar de agradecer o apoio estrutural da Faculdade de Tecnologia de Bauru, Fatec Bauru, que além de permitir que flexibilizasse meus horários, contribuiu grandemente para a realização das minhas pesquisas através de apresentações e participações em eventos e projetos. Sempre digo que a Fatec Bauru é minha segunda casa, e agradeço particularmente à direção, na pessoa do Prof. Dr. Sebastião Gândara, principalmente à minha Superiora imediata, Diretora de Serviços Administrativos Priscilla Mainini, além dos auxiliares docentes, Paulo Sérgio, Cleiton Carvalho e Rafael Balan que colaboraram grandemente no desenvolvimento técnico, eletrônico e prototipagem do dispositivo. Ainda quero agradecer o apoio de Professores essenciais para o desenvolvimento do “núcleo duro” do meu projeto, como o Professor Me. Claudines Torres e o Prof. Me. Adriano Mazotti, entre outros. Assim como à Faculdade Anhanguera de Bauru, seus professores e funcionários, e particularmente o Prof. Alexandre Nicolas, coordenador

de Ciências da Computação, que esteve em minha banca de graduação em 2007 e hoje é meu colega de trabalho.

O trabalho de outras pessoas foi essencial para a realização dessa etapa. Entre elas minha amiga Luciane Giroto, que sempre confiou (até mais que eu mesmo) em meu potencial, minha colega de orientação Renata Fakhoury, grande mestra das competências científicas e meu ex-estagiário e amigo Thiago César, meu grande incentivador no uso da linguagem Python e responsável por ótimas ideias (e códigos) durante o desenvolvimento do Software. Até meu pai participou das medições e testes, quando meus horários de trabalho não permitiam que eu estivesse em campo.

Além deles, gostaria de agradecer todos os colegas ingressantes da turma de 2017: Alessandro, Ivan, Nicolas, Brenda, Elaine e todos os outros, pelo companheirismo, pelas ideias e principalmente por compartilhar desse sonho de se tornar Mestre em Mídia e Tecnologia.



“Com um grande poder vêm também grande responsabilidade”

(Stan Lee - Uncle Ben. Amazing Fantasy, v. 15, 1962)

MIRA, J. E. **Desenvolvimento de um modelo de neurônio artificial com microcontrolador Wi-Fi autônomo com comunicação M2M para IoT**, 2019, 71 f. Trabalho de Conclusão (Mestrado Mídia e Tecnologia) - FAAC - UNESP, sob a orientação do Prof. Titular João Fernando Marar, Bauru, 2019.

RESUMO

Objetos conectados e um ambiente sensível capaz de interagir de forma calma e praticamente onipresente entre si e com humanos. Essa é a premissa da computação ubíqua, termo cunhado em 1991 pelo cientista da computação Mark Weiser. Recentemente têm-se ouvido falar de forma cada vez mais frequente de aplicações e dispositivos para internet das coisas, porém existem aplicações complexas demais para uso e outras têm custo proibitivo para implantação em larga escala. O atual projeto tem como objetivo investigar as potencialidades do desenvolvimento de um dispositivo para ser um modelo de neurônio artificial para Internet das Coisas e comunicação M2M utilizando-se do microcontrolador Wi-Fi ESP8266 e da linguagem de programação Micropython. Justificamos para a escolha desse microcontrolador, além do seu baixo custo e simplicidade, o fato de comparativamente ser uma opção mais prática e de baixo consumo de energia, quando comparado com outros dispositivos semelhantes, como Arduino e Raspberry Pi. Para nossa pesquisa utilizaremos além de pesquisa bibliográfica uma proposta prática de utilização do dispositivo com compartilhamento de informações de sensores através da rede wireless. A metodologia utilizada constitui-se na proposta do dispositivo, seu desenvolvimento e testes, bem como a demonstração dos scripts de programação utilizados no sistema.

Palavras-chave: computação ubíqua; IoT; microcontrolador; mídia e tecnologia; rede neural.

MIRA, J. E. **Desenvolvimento de um modelo de neurônio artificial com microcontrolador Wi-Fi autônomo com comunicação M2M para IoT**, 2019, 71 f. Trabalho de Conclusão (Mestrado Mídia e Tecnologia) - FAAC - UNESP, sob a orientação do Prof. Titular João Fernando Marar, Bauru, 2019.

ABSTRACT

Connected objects and a sensitive environment capable of interacting in a calm and practically omnipresent way with each other and with humans. This is the premise of ubiquitous computing, a term coined in 1991 by computer scientist Mark Weiser. Recently there has been a listen in an increasing frequency of applications and Internet devices of things, but there are too many applications that are too complex to use and others have a prohibitive cost for large-scale deployment. The current project aims to investigate the potential of developing a device to be an artificial neuron model for Internet of Things and M2M communication using the Wi-Fi ESP8266 microcontroller and the Micropython programming language. We justify for the choice of this microcontroller, in addition to its low cost and simplicity, the fact that it is comparatively a more practical and low energy option when compared to other similar devices, such as Arduino and Raspberry Pi. For our research we will use besides bibliographical research a practical proposal of use of the device with information sharing of sensors through the wireless network. The methodology used is based on the proposal of the device, its development and testing, as well as the demonstration of programming scripts used in the system.

Keywords: ubiquitous computing; IoT; microcontroller; media and technology; neural network

LISTA DE ILUSTRAÇÕES

FIGURA 01: Exemplo de uma rede neural comum, simulada.....	16
FIGURA 02: Um cenário para o teste de Turing.....	27
FIGURA 03: Módulo ESP-01 fabricado pela AI Thinker.....	32
FIGURA 04: O módulo ESP8266 fabricado pela chinesa Ai-Thinker.....	33
FIGURA 05: Pinos do microcontrolador ESP8266ex.....	34
FIGURA 06: Modelo de neurônio artificial.....	35
FIGURA 07: Modelo da rede de comunicação autônoma proposta com o módulo ESP8266.....	37
FIGURA 08: Uma das funções do código.....	38
FIGURA 09: Fluxograma completo de funcionamento do sistema.....	39
FIGURA 10: Bateria de 3,6v e 600 mah.....	41
FIGURA 11: Célula solar de 5v e 60 mah.....	43
FIGURA 12: Imagem de um protótipo enviado para impressão em 3D.....	48
FIGURA 13: Visualização dos dados no formato recebido pelo navegador.....	49
FIGURA 14: Navegador de internet Mozilla acessando os dados do node.....	50
FIGURA 15: Processo de transmissão dos dados através dos nodes.....	51
FIGURA 16: Dispositivo modular operando com três células de 5v e 60mah.....	52
FIGURA 17: Apresentação de Artigo no EBICC 2017.....	53
FIGURA 18: Apresentação de Pôster com tecnologia IOT do ESP8266 no Arduino Day da Fatec Bauru.....	54

SUMÁRIO

1 INTRODUÇÃO	12
1.1 Objeto e problema	13
1.2 Objetivo geral	14
1.3 Objetivos específicos	14
1.4 Justificativa	15
1.5 Materiais e métodos	16
1.6 Cronograma	19
2 DADOS GERAIS DO PRODUTO	20
3 PROPOSTA DE NEURÔNIO ARTIFICIAL SEM FIO	22
3.1 IoT, M2M e a computação ubíqua e pervasiva	22
3.2 Redes neurais e neurônios artificiais	26
3.2.1 Definições de Inteligência Artificial	27
3.2.2 Humanas x Exatas	29
3.3 Computação ubíqua e redes neurais	30
4 O MICROCONTROLADOR ESP8266 E A LINGUAGEM MICROPYTHON	32
4.1 O microcontrolador ESP8266 e suas características	32
4.2 Modelo teórico de uma rede neural baseada no módulo ESP8266	35
4.3 Sensores, números randômicos e células solares	40
4.3.1 Células solares e baterias	43
4.6 Bibliotecas	41
5 JUSTIFICATIVA	47
6 RESULTADOS OBTIDOS	49
7 OUTROS RESULTADOS OBTIDOS	53
7.1 Apresentação EBICC	53
7.2 Poster IoT	54

8 PARCERIAS INSTITUCIONAIS	54
9 IMPACTOS	55
9.1 Dispositivo autônomo	55
9.2 Prototipagem em 3D	55
9.3 Prototipagem em 3D	56
10 DIFICULDADES	56
10.1 Importação e documentação, hardware fechado	56
11 CONSIDERAÇÕES FINAIS	58
REFERÊNCIAS	63
APÊNDICES	67

1 INTRODUÇÃO

As possibilidades de um “admirável mundo novo” parecem estar mais próximas da realidade com a evolução da comunicação proativa de computadores sem a interferência humana, termo conhecido como M2M e significa *Machine to Machine*, ou comunicação de máquina para máquina. Essas comunicações nasceram “das tecnologias desenvolvidas para possibilitar a comunicação totalmente autônoma entre dispositivos e equipamentos sem qualquer intervenção humana” (GOUVEIA, 2013). Segundo um relatório recente da Harbor Research, “Esses serviços “invisíveis” serão muito mais importantes para os negócios - e para a evolução de nossa economia - do que os serviços complicados de hoje”. Mais abrangente e popular, é o conceito de Internet das Coisas, ou IoT (do inglês, *Internet of Things*). A definição de IoT está longe de ser ponto pacífico entre os autores. Alguns objetificam o meio, enquanto outros acreditam que também os objetos devem ser tratados como parte da infraestrutura da IoT (SINGER, 2015). Em nosso trabalho, tratamos não só a comunicação e os processos como parte da IoT, como também os objetos que se comunicam, na clássica dicotomia de hardware e software. Se logo todos os computadores serão equipados com alta capacidade de processamento e com habilidades de comunicação autônomas e proativas, é factível um cenário onde estes se tornaram capazes de começar a pensar horizontalmente, acessando informações e “sentindo” informações de pessoas, objetos e ambientes.

A ideia de comunicação entre objetos ganhou notoriedade científica recentemente com um crescente volume de artigos em revistas e eventos (SINGER, 2015), além de reportagens na mídia televisiva e escrita sobre a Internet das Coisas. IoT seria a migração da Internet como conhecemos hoje, como algo ao qual se conecta, para um ambiente nativo onde objetos (e futuramente, pessoas) estariam nativamente conectados, interagindo de maneira natural e pervasiva (GOUVEIA, 2013). Se por um lado, a IoT permite a comunicação de objetos com outros objetos, servidores e serviços, a M2M sugere uma rede capilarizada, onde cada objeto se conecta com o próximo e assim sucessivamente. Computação ubíqua é um termo apresentado em um artigo seminal por Weiser (1991), e citado por Saccol e Reinhard (2007) sobre a absorção dos computadores pelos ambientes, de maneira onipresente.

Quando os primeiros artigos sobre pervasivos e ubíquos foram escritos, não eram populares (e alguns casos sequer existiam fora dos laboratórios de pesquisas) alguns equipamentos, protocolos e soluções que hoje são comuns em nosso dia a dia. Dois exemplos disso são os smartphones e a computação em nuvem (*cloud computing*). Considerando-se essas possibilidades de objetos e ambientes amplamente conectados, propõe-se uma pesquisa que apresenta um dispositivo programado com uma linguagem específica para tornar factível aplicações para IoT que seja de fácil aplicação e modificação, utilizando código aberto e ferramentas open-source bem como a possibilidade de se utilizar poucos recursos energéticos e financeiros¹.

1.1 Objeto e problema

Para o desenvolvimento da proposta estabelecida neste tema, durante a realização deste trabalho utilizaremos o módulo ESP8266, um microcontrolador sem fio, em uma versão conhecida como ESP12 e vendida comercialmente como NodeMCU, embora o nome NodeMCU faça referência ao *firmware*² padrão, que não será utilizado nesta proposta (NODEMCU COMPANY, 2017). Este módulo pode se conectar diretamente às redes wireless padrão 802.11, que são normalmente disponibilizadas para pontos de acesso à internet sem fio. Ele atua tanto como um ponto de acesso (modo AP) quanto um cliente (modo STA), possibilitando assim a criação de redes ad-hoc (ANISH; KIRUBAKARAN, 2016) com alcances maiores do que as oferecidas pela tecnologia *Bluetooth*, por exemplo. Além do alcance maior, a possibilidade de integração nas redes 802.11³ permitiria na integração do módulo a redes já existentes, abrindo assim um leque de opções ainda maior, como encontrar e se conectar em redes Wi-Fi, inclusive funcionando simultaneamente como cliente e ponto de acesso.

A comunicação dos sensores e interação com outros dispositivos é feita por portas seriais, e a programação se dá através de *scripts* personalizados carregados através de uma versão de *firmware* específica com uso das portas GPIO, ou *General*

¹ Parte desse capítulo texto foi livremente baseado e adaptado de um artigo apresentado no II Congresso de Iniciação Científica da Fatec Lins com o título “Proposta teórica de uma rede neural sem fio baseado no módulo WiFi ESP8266”.

² O conceito de firmware e suas características é mais bem abordado no item 1.5, materiais e métodos.

³ Em referência ao protocolo 802.11 da Institute of Electrical and Electronic Engineers (IEEE) que estabelece os padrões de utilização de redes Wi-Fi.

Purpose Input/Output, que basicamente são portas programáveis de entrada e saída e dados, utilizadas para prover uma interface entre os periféricos e os microprocessadores (ESPRESSIF, 2014). Dois aspectos chamam muito a atenção a favor deste módulo: o tamanho reduzido e o baixo custo. Outro aspecto é seu baixo consumo de energia elétrica e a possibilidade de, com a utilização de bibliotecas específicas de seus sistemas de *firmware*, a utilização em modo intermitente de consumo de energia, a utilização de uma bateria recarregável a um sistema de carga baseado no uso de células solares.

Neste trabalho, pretende-se utilizar a linguagem de programação Micropython e algumas dessas suas bibliotecas, além de outras ferramentas de código aberto, de acordo com o disposto na seção sobre metodologia. O objeto da pesquisa será, portanto, a proposta de um dispositivo sustentável para aplicações de IoT com a utilização do ESP8266 e Micropython. Seguindo essa proposta, desenvolver-se-á uma proposta de utilização desse microcontrolador com a utilização das bibliotecas do Micropython para desenvolvimento de um dispositivo autônomo de funcionamento análogo ao neurônio, baseado no modelo de neurônio artificial através de comunicação M2M em aplicações de IoT, em ambientes e aplicações diversas, de acordo com alguns trabalhos já publicados (MIRA; MARAR, 2017). Tanto o microcontrolador quanto a linguagem tem suas especificidades e características explicadas de forma mais pormenorizada na seção 1.5 de Materiais e Métodos.

1.2 Objetivo geral

A finalidade do projeto é a utilização do microcontrolador ESP8266 no modelo ESP12 (NodeMCU) com o software Micropython em um sistema de comunicação M2M com funcionamento análogo ao do neurônio, para utilização em IoT, para a leitura de sinal de sensores, com baixa utilização de energia elétrica, além do uso de uma fonte de energia alternativa, no caso, células solares.

1.3 Objetivos específicos

a) Descrever as características do ESP8266 e a possibilidade de utilização da linguagem de programação Micropython e suas principais bibliotecas para uso associada ao microcontrolador ESP8266;

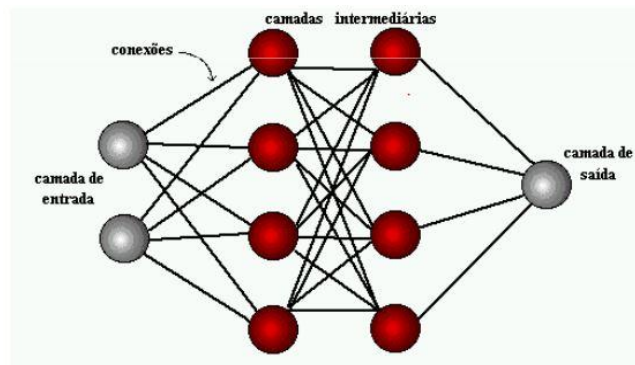
b) Propor um modelo de sistema de funcionamento análogo ao neurônio artificial sem fio através da utilização de um microcontrolador wireless.

1.4 Justificativa

A escolha do objeto da pesquisa deu-se após um processo de pesquisa prévio que antecede, inclusive, o desenvolvimento deste projeto. Ao procurar um microcontrolador para um sistema simplificado de controle de presença acionado via internet para veículos em 2013, foi observado que a utilização de uma placa Arduino exigia um grande consumo de energia mesmo quando o sistema não estivesse em operação. Além do consumo elétrico, a placa Arduino era grande e tinha muitas opções de utilização que ficariam inativas, ao passo que para uma das funções principais, a conectividade à internet, seria necessária a utilização de uma placa externa de rede, conhecida como Shield Ethernet. A pesquisa que se realizou a partir daí mostrou que o ESP8266 possuía, além das características de controle de sinais através das portas digitais e analógicas, leitura de programas carregados em forma de *script* e baixo consumo de energia, uma interface de rede sem fio embutida que poderia conectá-lo a internet e ao mesmo tempo torná-lo um dispositivo que funcionasse como um servidor, permitindo que outros equipamentos se conectassem à ele.

A partir daí, começou a desenhar-se a possibilidade da utilização de um microcontrolador para criação de um dispositivo que funcionasse de forma análoga ao neurônio, criando a possibilidade de uma rede neural sem fio baseada em dispositivos autônomos, capaz de receber sinais do ambiente externo, através de sensores, tratá-los matematicamente e enviá-los, através de uma rede de conexão WiFi, de forma semelhante às sinapses humanas. Um sistema assim apresentaria vantagens em relação a outros sistemas de comportamento de simulação de rede neural, que se comunicam em neurônios emulados fisicamente dentro de um software ou hardware, conforme está ilustrada na Figura 01. Nesta proposta, efetivamente cada um dos dispositivos seria um neurônio, autônomo, porém não isolado.

Figura 01 - Exemplo de rede neural comum, simulada.



Fonte: PEREIRA (2008)

Essas características foram depois somadas as vantagens da utilização de um firmware que permitia a utilização de uma linguagem popular de código aberto, o Micropython, com as possibilidades de utilização de algumas de suas bibliotecas para aplicações de baixo custo para M2M e IoT. No entanto, na literatura e na academia existem poucas pesquisas que enfoquem a utilização do ESP8266 em conjunto com o Micropython para aplicações de monitoramento ou de IoT⁴.

1.5 Materiais e métodos

Para esse trabalho, será utilizado como microcontrolador o módulo Wi-Fi ESP8266, conforme descrito no item 1.2. O espaço interno de alocação para *firmware* e *scripts* varia de módulo para módulo uma vez que o ESP8266 possui diversos modelos de hardware com diferenças de tamanho, quantidade de pinos de entrada e tamanho de memória *flash*. No módulo que foi utilizado na proposta a memória *flash* interna é de 4MB, espaço que atende com folga as necessidades do projeto para a inserção dos arquivos que serão utilizados na proposta.

A ideia de utilizar o microcontrolador ESP8266 surge, principalmente do comparativo com outras opções disponíveis. Essas outras opções apresentam todas as características presentes nos objetivos da computação ubíqua como plataformas abertas e simples, que permitam o reconhecimento de padrões ambientais e

⁴ Encontramos alguns textos associando o uso do ESP8266 com a utilização da linguagem Micropython, a maioria para aplicações domésticas, na China e na Índia. Um dos principais resultados para os termos em um artigo nacional apresentado em um Congresso de Ensino e Extensão em 2017 no IFRS indica que os autores preferiram utilizar a linguagem Lua ao invés do Micropython.

controle de cargas elétricas e mecânicas, além da transmissão desses dados com protocolos seguros e comuns.

Uma delas, possivelmente a mais popular, é chamada de plataforma de prototipagem Arduino (MCROBERTS, 2011). Segundo a página oficial do projeto, “Placas Arduino estão aptas a ler entradas [...] e transformar isso em uma saída [...]. Tudo isso é definido por um conjunto de instruções programadas através do Software do Arduino (IDE)”. IDE é um acrônimo de *Integrated Development Environment*, ou Ambiente de Desenvolvimento Integrado, que consiste em um editor de código, uma área de mensagens para *debug*, um console de texto, uma barra de ferramentas com botões para as funções comuns e alguns menus, além de um sistema de interface serial para leitura em tempo real de informações na porta serial e a conexão para carregamento dos programas na memória do programa. A placa em si consiste em um microcontrolador Atmel AVR de 8 bits, um cristal oscilador para o *clock*, um regulador linear de 5 volts e, em alguns casos, uma saída USB para conexão ao computador. Os pinos I/O do microcontrolador são expostos para que possam ser utilizados. (MCROBERTS, 2011), embora tenha diversas características (consumo de energia do arduino e do ESP).

Uma outra opção que tem se tornado popular para soluções de hardware e software para aplicações embarcadas e de baixo custo é a plataforma Raspberry Pi. Com capacidade computacional acima dos microcontroladores supramencionados, ela tem características que o encaixam melhor como um microcomputador do que propriamente um microcontrolador. Ainda no prefácio do livro *Getting Started with Raspberry Pi* (RICHARDSON; WALLACE, 2016), os autores dizem que o Raspberry não se diferencia de um computador normal apenas pelo tamanho e preço, mas principalmente pela sua capacidade de se integrar com outros projetos eletrônicos. Os autores dizem claramente que o Raspberry Pi não é melhor que um microcontrolador, apenas diferente, mas que ele tem mais em comum com um computador (pessoal) do que com um microcontrolador Arduino, por exemplo.

É importante salientar que todos os softwares e códigos utilizados durante a execução dos testes⁵ e do desenvolvimento desta proposta são de código aberto

⁵ Os testes realizados durante o desenvolvimento da proposta são descritos no Capítulo 04.

e/ou GNU. Utilizou-se um *firmware* específico desenvolvido pela George Robotics Limited e licenciado sobre uma licença Open Source. A gravação do *firmware* foi realizada com o Software Esptools (GRATTON, 2015), a cópia dos arquivos com o Ampy (DICOLA, 2014) e a edição do código é realizada com o editor de textos avançado Kate, todos eles disponíveis para download gratuito através do GitHub dos seus desenvolvedores. Já a programação em si utiliza a linguagem específica de microcontroladores Micropython, uma implementação leve e eficiente da linguagem de programação Python 3 com um pequeno subconjunto de sua biblioteca padrão e otimizada para execução em microcontroladores e outros ambientes restritos (GEORGE ROBOTICS LIMITED, 2014). As principais bibliotecas utilizadas terão a função de controlar a entrada e saída de sinais nos pinos do ESP8266, disponibilizar esses dados pela interface de rede através de um protocolo comum e controlar a utilização da energia elétrica nesses pelo microcontrolador.

Parte importante do processo de utilização do Micropython é a gravação de um novo *firmware*, que consiste na substituição do sistema original pelo *firmware* Open Source, através de uma gravação realizada com um chip FTDI, o FT232R da SparkFun Electronics. Este chip é um conversor USB – UART⁶ de fácil utilização, que uma vez conectado aos pino TX e RX do ESP8266, pode realizar a limpeza do sistema antigo da memória (os microcontroladores ESP8266 são equipados de fábrica com um sistema que atende comandos do tipo AT, um padrão de comunicação para *modems* e outros equipamentos de transmissão) e os substituir pelo *firmware*, que no caso é a versão 1.9.3, disponibiliza apenas para placas com 1024kb de memória⁷. Na versão final do projeto utilizou-se um modelo do ESP-12 que já possui um chip conversor USB-UART integrado, o que facilitou o processo de gravação de arquivos e testes constantes durante o desenvolvimento, porém durante os primeiros testes foi utilizado um chip de gravação modular. Ambos funcionam de maneira exatamente análoga⁸.

⁶ De acordo com o dicionário de Computação Oxford, UART é acrônimo de Universal Asynchronous receiver/transmitter, ou Transmissor/Receptor Assíncrono Universal, um circuito lógico capaz de converter bytes em paralelo em uma sequência de bytes seriais assíncronos e vice e versa. (2008)

⁷ Parte dessa descrição do Chip UART integra um de nossos trabalhos enviados para publicação.

⁸ Conforme descrito na seção 4.1.

1.6 Cronograma

Abaixo, um quadro que ilustra uma proposta de organização das atividades propostas para a realização da pesquisa no período compreendido entre março de 2017 e março de 2019.

Quadro 01 – Cronograma das atividades

	1º Semestre 2017	2º Semestre 2017	1º Semestre 2018	2º Semestre 2018
Cursar disciplinas	X	X	X	X
Reunir material de leitura e fichamentos	X	X	X	
Leitura ativa e escrita do projeto		X	X	
Elaborar a proposta do dispositivo		X	X	
Testes de célula de alimentação solar			X	X
Desenvolvimento da versão final do software			X	X
Teste de conexão entre os dispositivos em rede			X	X
Escrever o relatório		X	X	X

Fonte: Próprio autor (2019)

2 DADOS GERAIS DO PRODUTO

Nome do Produto: SinESP 1.0

Período de Execução Física: Entre agosto de 2017 à janeiro de 2019

Valor total do projeto (incluindo todos os intervenientes):

Quadro 02 - Componentes e intervenientes utilizados

Nome	Descrição	Preço (R\$) ⁹
ESP8266 - ESP12 NodeMCU (Fabricado pela Lolln)	Módulo do microcontrolador utilizado na versão final (3 unidades)	10,69
Sensor DHT 22	Sensor de temperatura e umidade	10,89
ESP8266 - ESP01	Não utilizado na versão final ¹⁰	6,62
Células solares mini PET 5v 60ma	Células solares (06 unidades)	20,00
Bateria	Bateria Ni-Cd 3,6v 600mah	20,12 ¹¹
Conversor FTDI	Gravador serial utilizado com o ESP-01	6,86
Osciloscópio	Utilizado para análise do sinal do sensor ¹²	55,55
Total		R\$ 130,73

Fonte: Próprio autor (2019)

⁹ Valores cotados em 22 de novembro de 2018. Todos os itens foram importados da China e tem portanto seu preço em dólar.

¹⁰ Conforme descrito no item 4.1.

¹¹ Único item adquirido no Brasil.

¹² Sensor não utilizado na versão final.

Bolsas - Financiamentos – Convênios e Parcerias (descrever o tipo de apoio recebido): N/A

Instituições participantes (nomear): Fatec Bauru (Participação no evento Arduino Day e uso de equipamentos da instituição)

Caracterização da Pesquisa: PESQUISA TÉCNICO-CIENTÍFICA - BASE TECNOLÓGICA

Caracterização da pesquisa com uma breve justificativa para o enquadramento: A pesquisa com o dispositivo pressupõe conhecimentos técnicos anteriores, como conhecimentos básicos de eletrônica e programação, procedimentos de gravação de firmware e sistemas de carga simples, entre outros. Como se pressupõe a ser uma ferramenta para a solução de alguns problemas como a comunicação sem fio entre dispositivos isolados fisicamente em áreas sem conexão à internet, considera-se uma pesquisa de base tecnológica.

3 PROPOSTA DE NEURÔNIO ARTIFICIAL SEM FIO

3.1 IoT, M2M e a computação ubíqua e pervasiva

Logo nos primeiros anos da década de 90, Mark Weiser dizia em artigo sobre a computação do século XXI que a disposição de interfaces de uso era apenas um estado transicional da computação, e que não representava seu real potencial. Weiser (1991) disse ainda que em breve a computação seria invisível e onipresente. A ideia de comunicação entre objetos ganhou notoriedade científica recentemente com um crescente volume de artigos em revistas e eventos (SINGER, 2012), além de reportagens na mídia televisiva e escrita sobre o tema.

Tais possibilidades de um novo mundo aproximam-se da realidade com a evolução da comunicação proativa de computadores independentemente da interferência humana. A comunicação direta de computadores entre si, é conhecido como M2M que significa *Machine to Machine*, ou comunicação de máquina para máquina.

Machine-to-Machine (M2M, também conhecido como a internet das coisas) refere-se às tecnologias que permitem ambas tecnologias com ou sem fio a se comunicarem com outros dispositivos que possuam a mesma habilidade. M2M usa um dispositivo (como um sensor ou medidor) para capturar um evento (como a temperatura, o nível de estoque entre outros), que é enviado por meio de uma rede (sem fio, com fio ou híbrida) para uma aplicação (programa), que transforma o evento capturado em informação útil (PINOCHET, 2014, p. 201).

Essas comunicações nasceram, segundo Gouveia, (2013), a partir de tecnologias desenvolvidas com a finalidade de possibilitar a comunicação totalmente autônoma “entre dispositivos e equipamentos sem qualquer intervenção humana” (GOUVEIA, 2013). Segundo um relatório recente da Harbor Research, “esses serviços “invisíveis” serão muito mais importantes para os negócios - e para a evolução de nossa economia - do que os serviços complicados de hoje”. Ainda mais abrangente e popular, é o conceito de Internet das Coisas, ou IoT (do inglês, *Internet of Things*), apontada como uma das principais tecnologias disruptivas com impactos potenciais no Estados Unidos de acordo com o relatório anual do Conselho Nacional de Inteligência, em 2008.

A Internet das Coisas (em inglês, Internet of Things – IoT) é a extensão para o mundo real, físico, da internet convencional. Resumidamente, na internet convencional, as mais diferentes informações estão armazenadas em computadores ao redor do mundo. Trata-se de um mundo completamente virtual em que se navega ou se interage com páginas acessadas por meio de hiperlinks. Na IoT, os objetos são identificados e tal identificação pode ser lida por meios automatizados. A partir daí os objetos físicos passam a ter uma representação no meio virtual (AMAZONAS, 2013, p. 57).

Nessa discussão temos autores que relacionam o apenas o meio ao termo, ao passo que outros acreditam que também os objetos devem ser tratados como parte da infraestrutura da IoT (SINGER, 2015). Se em breve todos os computadores serão equipados com alta capacidade de processamento e com habilidades de comunicação autônomas e proativas, então é possível um cenário onde estes se tornaram capaz de começar a pensar de forma linear, horizontal, acessando informações e percebendo informações de pessoas, objetos e ambientes.

Internet das Coisas (IoT) seria então essa migração da Internet como existe hoje, um ambiente ao qual se conecta, para um algo mais abstrato, uma rede de comunicação onipresente onde objetos e futuramente até pessoas estariam nativa e constantemente conectados, interagindo de maneira natural e pervasiva. Gouveia (2013) defende ainda que o objetivo da IoT é principalmente a troca de informação relevante, informação essa proveniente dos objetos e dispositivos que tenham capacidade de processamento de dados, armazenamento e de comunicação com outros dispositivos.

Definir o que é a Internet das Coisas parece ainda mais difícil quando olhamos para a quantidade de assuntos que a temática pode envolver, [...]. Nas publicações existem artigos sobre inteligência espacial, coleta de dados, sensores de baixo consumo de energia, middleware, segurança de rede, criptografia, design centrado no usuário, arquitetura de informação (SINGER, 2012, p. 3).

Objetos conectados recebem e transmitem dados via internet, especificando suas preferências e controlando-os remotamente, em tempo real (WEINBERG et al., 2015). A IoT coleta e gerencia dados de uma rede de sensores e dispositivos, processa-os e os compartilha com outras redes conectadas e seu domínio dos ambientes é visto como um elemento-chave para seu futuro (JACOBSSON, DAVIDSSON; 2015).

Se por um lado a IoT permite a comunicação de objetos com outros objetos, servidores e serviços, a M2M sugere uma rede capilarizada, onde cada objeto se conecta com o próximo e assim sucessivamente. Nascida a partir das tecnologias que foram excepcionalmente desenvolvidas para possibilitar a comunicação totalmente autônoma entre dispositivos e equipamentos, sem que houvesse qualquer intervenção humana (GOUVEIA, 2013) ¹³. Ambos os conceitos, separadamente ou combinados, são elementares para o que se chama de computação ubíqua e computação pervasiva. Apesar disso, neste trabalho será tratado sobre a M2M como um conceito implícito no conceito de IoT.

Quando os primeiros artigos sobre pervasivos e ubíquos foram escritos, não eram populares alguns equipamentos, protocolos e soluções que hoje são comuns, no qual casos relacionados ao tema sequer existiam fora dos laboratórios de pesquisas. Dois exemplos disso são os smartphones e a computação em nuvem (*cloud computing*). Hoje, no entanto, já se encontra uma bibliografia extensa sobre uso e aplicações, que vem a confirmar o que Weiser (1991) dizia sobre a tecnologia necessária para a computação onipresente ser composta por três partes: computadores de baixo consumo de energia e baratos com telas convenientes, uma rede para conectá-los e softwares implementação de aplicativos onipresente. Atualmente, tais equipamentos já estão disponíveis. Além dos autores já citados nesse parágrafo, como Gouveia, outros como Santaella e Bradley também abordaram o assunto. Além da discussão filosófica, que trata dos fins para o qual se busca uma inteligência através da comunicação independente de objetos, há também a discussão acerca dos meios. Sobre isso falaremos mais nos capítulos de desenvolvimento do dispositivo¹⁴.

Sobre as características gerais da utilização, Atzori et al. (2010) sistematiza o conhecimento sobre IoT e suas características, principalmente no que concerne a relação com as pessoas à potenciais implicações de segurança, conforme pode ser visto no fragmento abaixo:

¹³ Trecho presente na introdução, por relevância da citação.

¹⁴ Presente no capítulo 4.

A Internet mudou drasticamente a maneira como vivemos, movendo as interações entre as pessoas em um nível virtual em diversos contextos, desde a vida profissional até as relações sociais. A IoT tem o potencial de adicionar uma nova dimensão a esse processo, permitindo comunicações com e entre objetos inteligentes, levando assim à visão de comunicações "a qualquer momento, em qualquer lugar, qualquer mídia, qualquer coisa (ATZORI et al., 2010, p. 16).

Ainda sobre as possibilidades de utilização da IoT e seu potencial, um outro autor pontua sobre como a relação "homem-máquina-espaco" será modificada com a introdução da IoT. Segundo Santaella (2008), computação móvel e pervasiva (referente à existência de computadores em todos os lugares) é a chave para a compreensão das mídias locativas, tecnologias essas que teriam como principal desafio a possibilidade de transmitir informação geográfica não mais nos desktops, mas nos sistemas móveis, aumentando assim a importância dos itens de IoT para além da comunicação, como uma forma de compreensão e utilização do próprio espaço físico. A abordagem proposta para implementar um sistema de IoT para criação de valor sustentável consistiria então na alavancagem do conhecimento de domínio e/ou do especialista de um produto, processo ou de um sistema por meio de projeto de hardware do sistema paralelo e da arquitetura da IoT (BRADLEY et al., 2017). Santaella vai ainda além: ao citar Weiser, ele o faz no contexto dos objetos cada vez mais disponíveis e rastreáveis, e cada vez menos visíveis. O smartphone que hoje está no bolso, longe das vistas, um dia foi um grande desktop bege em uma mesa de escritório, e antes, um mainframe ruidoso em alguma sala refrigerada de concreto armado. Na contramão da suntuosidade, os dispositivos se tornaram cada vez mais potentes, em poder de armazenamento e processamento de informação, e mais recentemente, na capacidade de comunicação.

Por meio de dispositivos dedicados, os computadores vão gradativamente sumir da nossa vista, enquanto as habilidades de processamento de informação vão emergir por todo o ambiente circundante. Com a capacidade de processamento de informação integrada, os produtos vão possuir habilidades de inteligência. Eles poderão também adquirir identidades eletrônicas que podem ser pesquisadas remotamente ou serem equipados com sensores para detectar mudanças físicas no seu entorno. (SANTAELLA, 2008)

O artigo seminal citado pelo autor, Computador para o século XXI, inaugurou a ideia de computação ubíqua, um termo apresentado por Weiser (1991), e citado por Saccol e Reinhard (2007) sobre a absorção dos computadores pelos ambientes, de maneira onipresente. Assim, levando-se em conta um cenário futuro de objetos e ambientes conectados de forma ampla, apresentamos o desenvolvimento de um

dispositivo programado com uma linguagem aberta baseado em um microcontrolador de baixo custo, com a proposta de simular de forma simplificada o processo de comunicação que ocorre no neurônio através das sinapses, guiados pela possibilidade de pouco consumo de recursos, tanto energéticos quanto financeiros.

3.2 Redes neurais e neurônios artificiais

Redes neurais artificiais são baseadas no modelo do neurônio humano e sua capacidade de tomada de decisão através de entradas, comparações e saídas, modelando assim amplas possibilidades de construção de modelos para comparação e respostas e também aprendizado através de sensoriamento, conforme já tratamos anteriormente (MIRA; MARAR, 2017)¹⁵.

Esse capítulo expõe e discute algumas questões relativas à inteligência artificial e à computação cognitiva e sua relação com a comunicação dos humanos. Entre os assuntos abordados estarão as definições literárias e clássicas de inteligência, os tipos de inteligência e a definição de sistemas considerados inteligentes, bem como as diferenciações relativas ao pensamento humano e pensamento de máquina.

Algumas definições encontradas na literatura descritiva, como no dicionário Michaelis (1998) tratam de três definições clássicas de inteligência: a teológica diz que inteligência seria “um dom divino que nos tornaria semelhantes ao criador”, a filosófica por sua vez diz que seria “um princípio abstrato que é a fonte de toda a intelectualidade” e a psicológica que inteligência seria a capacidade de “resolver problemas novos com rapidez e êxito”¹⁶. De qualquer maneira, para fins de inteligência artificial e suas possibilidades se podem ater na última definição de que, em outras palavras, ser inteligente é a capacidade de resolver problemas novos. No entanto, uma máquina capaz de resolver um problema novo, por exemplo, uma equação, poderia ser considerada pensante?

¹⁵ Parte desse conteúdo foi desenvolvido durante a disciplina de Cognição e sistemas computacionais inteligentes em TV Digital e apresentado posteriormente no congresso da USP, o EBICC 2017

¹⁶ Livre transcrição do dicionário Michaelis de 1998, e considerada senso comum.

3.2.1 Definições de Inteligência Artificial

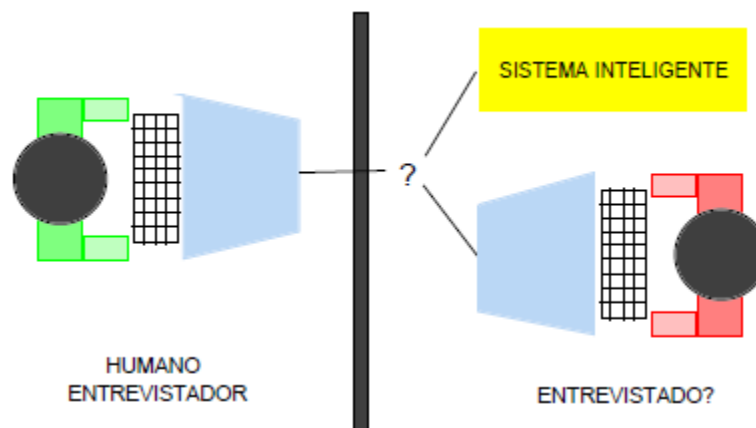
Russel e Norwig (1995) classificam em quatro as definições de IA encontradas na literatura científica, a saber:

- (a) sistemas que pensam como humanos;
- (b) sistemas que agem como humanos;
- (c) sistemas que pensam logicamente;
- (d) sistemas que agem logicamente.

Segundo Pereira (2008), as duas primeiras categorias, a e b, são, essencialmente empíricas e envolvem formulação de hipóteses e confirmação experimental. Já as outras duas c e d são teóricas e envolvem matemática e engenharia. O autor defende ainda que o pensar, de acordo com a definição clássica de Turing seria uma “imitação” do comportamento humano. Ora, essa é uma visão essencialmente antropocentrista, pois exige da máquina pensante um padrão de pensamento igual ao dos seres humanos, ao exigir que para a máquina “pensar como humano” deveria imitá-lo, passando-se por um.

O teste de Turing, proposto por Alan Turing (1950), foi projetado para fornecer uma definição operacional satisfatória de inteligência. O computador passará no teste se um interrogador humano, depois de propor algumas perguntas por escrito, não conseguir descobrir se as respostas escritas vêm de uma pessoa ou de um computador (RUSSEL; NORWING, 1995, PG 05).

Figura 02 – Um cenário para o teste de Turing



Fonte: Pereira (2008)

Apesar disso a definição pura de inteligência de Turing ainda se encontra com a descrita pela literatura de psicologia, ao defender que pensar seria a capacidade de processar informações ao ponto de resolver problemas. Russel e Norwig defendiam que haveria três formas de se determinar como os humanos pensam, penetrando nos componentes reais da mente humana: Através da introspecção, procurando captar os nossos pensamentos de forma particular, através de experimentos psicológicos e sua observação e a através de imagens cerebrais, observando o cérebro em ação. Turing (1950) é mais incisivo ainda, ao propor que não seria possível realizar uma imitação do sistema nervoso, e tão somente do comportamento do ser pensante:

O sistema nervoso certamente não é uma máquina de estado discreto¹⁷. Um pequeno erro na informação sobre o tamanho de um impulso nervoso que incide sobre um neurônio pode fazer uma grande diferença no tamanho do impulso de saída. Pode-se argumentar que, sendo assim, não se pode esperar ser capaz de imitar o comportamento do sistema nervoso com um sistema discreto (TURING, 1950, PG 15).

Nesta abordagem, porém, compreende-se que ao propor um modelo de neurônio sem fio dá-se um passo tímido no sentido de replicar¹⁸ parte da estrutura biológica que permite a cognição humana, imitando o comportamento das sinapses com envio e recebimento de dados através de uma rede sem fio, usando a dispersão. Ao replicar a menor partícula do sistema nervoso dos seres humanos, anseia-se alcançar o estado da arte onde um ecossistema neural baseado em módulos que se conectam através de uma rede sem fio seja possível.

O dilema aqui é saber se um problema resolvido com lógica formal e mecanicista teria o mesmo valor pensante de um problema resolvido com modelos mentais e estratégicas, de acordo com o que Johnson e Laird (2004) explicaram, onde as pessoas usariam dados conhecidos, experiências e conhecimento geral do mundo para “testar” o possível estado de todas as coisas do mundo, em um estado de abstração, onde se aplicaria uma estratégia para tentar se chegar a uma solução.

¹⁷ Ainda segundo Turing, máquinas de estado discreto era uma definição generalista para máquinas cujo funcionamento dá-se através de posições bem definidas e pré-determinadas, como os computadores binários modernos e suas portas lógicas

¹⁸ No sentido de copiar o comportamento, afim de simular o funcionamento. Pensamos no neurônio como hardware biológico

3.2.2 Humanas x Exatas

Para ilustrar a relação entre o pensamento baseado na lógica mecanicista e nos modelos mentais pode-se recorrer a dois campos das ciências que estudam áreas de interesses diferentes da sociedade humana, as ciências exatas e as ciências humanas.

Enquanto os primeiros computadores analógicos de que se têm notícias eram facilmente capazes de serem configurados para a solução de equações matemáticas complexas, ainda hoje esses sistemas computacionais apresentam dificuldade na elaboração de músicas, artes plásticas, composição e interpretação literária. O motivo disso é que o computador é uma mente solitária enquanto o ser humano tem, graças aos modelos mentais, um pensamento horizontal, que permite retirar de conhecimentos passados, sentimentos, sensações e até mesmo memórias pessoais informações que o ajudem a resolver um problema.

Alguns cientistas como Stephen Hawking já se manifestaram temerosos que uma inteligência artificial desperte em alguns momentos nos próximos anos, e torne-se onisciente. Esse evento é chamado pelos especialistas de Singularidade, e o professor João de Fernandes Teixeira em um artigo de filosofia intitulado “Filosofia da mente: A ética da singularidade” faz uma breve explicação sobre sua origem:

O termo singularidade foi cunhado por um autor de ficção científica, Vernor Vinge, num artigo publicado em 1983, intitulado *First Word*, que na época não teve muita repercussão. Recentemente, o termo foi apropriado pelo inventor e futurólogo Ray Kurzweil, que o usou profusamente nos livros *A singularidade está próxima* e *A era das máquinas espirituais*. Foi esse último uso que conferiu ao termo um suposto caráter futuroológico, que faz hoje filósofos duvidarem de sua seriedade (TEIXEIRA, 2016¹⁹).

Se logo todos os computadores serão equipados com alta capacidade de processamento e com habilidades de comunicação autônomas e proativas, é factível um cenário onde estes se tornaram capazes de começar a pensar horizontalmente, acessando informações e “sentindo” aquilo que os outros computadores sentem.

¹⁹ Essa referência está no blog do autor, mas foi posteriormente publicada no seu livro “O cérebro e o robô”, da editora Paulus. O texto abre o capítulo III.

3.3 Computação ubíqua e redes neurais

A exploração das opções não se exaure aos temas acima relacionados, e as possibilidades de adaptação, combinação e expansão dos projetos que podem ser realizados com o dispositivo baseado no ESP8266 são tão amplas como as combinações de peças de LEGO. Em um evento recente, foi apresentado um pôster equipado com um ESP8266 de como as pessoas poderiam interagir com um objeto, até então inanimado, e as possibilidades de envolvimento e publicidade que isso poderia envolver. O que é isso senão, a possibilidade de criar e/ou aumentar as emoções causadas por um produto? Certamente, produtos e publicidades que interagem entre si e com outros humanos têm uma chance efetivamente de gerar processos emocionais nas pessoas, conforme demonstra a conclusão de um trabalho conduzido por Scolari e Marar (2009).

O dispositivo proposto, apresentado como SinESP tem por objetivo ser uma plataforma de leitura de dados através de sensores de diversos tipos, de acordo com as necessidades da aplicação, e promover a replicação desses dados de forma semelhante ao neurônio, recebendo informações, realizando médias baseadas em pesos pré-estabelecidos em seu sistema interno e replicando para os demais dispositivos conectados na rede, de forma que esses dados podem ser aproveitados pelo próprio nó ou simplesmente encaminhados para um outro sistema. Além disso, o dispositivo tem características que dão a ele certa autonomia, como a capacidade de poder ao mesmo tempo receber conexões e se conectar a outros, ser alimentado por uma bateria e recarregado através de células solares.

Considerando o amplo espectro de possibilidades que podem ser desenvolvidas no âmbito da inteligência artificial e sistemas computacionais que emulam e/ou imitam o funcionamento da inteligência humana e as possibilidades de utilização do domínio da IoT no desenvolvimento de aplicações que sejam simultaneamente tecnológicas e compatíveis com os novos paradigmas ambientais de baixo consumo de recursos e ampla gama de comunicação, determinou-se que o tema deste projeto será o desenvolvimento de um modelo de neurônio baseado em

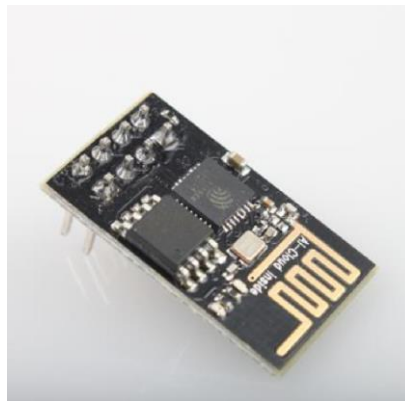
um microcontroladore e linguagem de programação específica para aplicações de IoT com comunicação M2M.

4 O MICROCONTROLADOR ESP8266 E A LINGUAGEM MICROPYTHON

4.1 O microcontrolador ESP8266 e suas características

O processo de desenvolvimento iniciou-se em 2017 com o módulo conhecido como ESP-01, fabricado pela empresa chinesa AI Thinker, (Figura 04). Embora atendesse as características técnicas de hardware desejadas, como a capacidade de operar simultaneamente como Access Point e estação WiFi e possuir pelo menos duas portas GPIO (para a posterior conexão física dos sensores), além de ser fisicamente menor do que os outros módulos, pois a interface de gravação FTDI não era integrada ao módulo²⁰. Apesar disso, diversas tentativas de gravação do *firmware* para a utilização da linguagem Micropython falharam, erros esses provavelmente associados ao fato dos módulos ESP-01 utilizarem memória interna de 512 KB (AI THINKER, 2015), o que impossibilitaria a utilização, uma vez que mesmo as versões mais antigas do firmware utilizados tinham tamanhos acima dos 450Kb, o que ocuparia praticamente toda a memória do módulo.

Figura 03 - Módulo ESP-01 fabricado pela AI-Thinker.



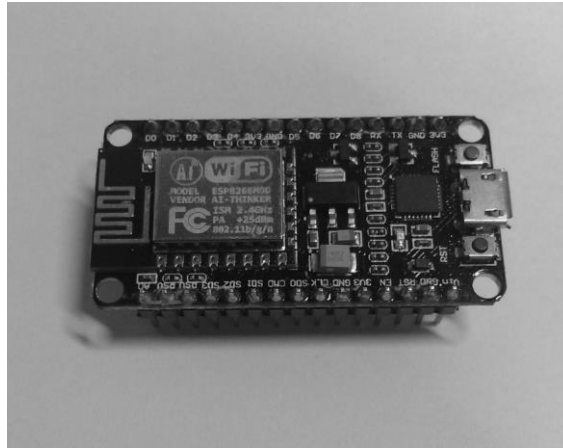
Fonte: AI-THINKER (2015)

Neste trabalho optou-se por utilizar uma das versões conhecidas do módulo wi-fi ESP8266, com 16 portas de entrada e saída em uma placa de programação desenvolvida para facilitar a carga de novas versões de *firmware*. Utilizou-se o modelo fabricado pela Amica e também uma placa paralela fabricada Wemos, o

²⁰ O processo de gravação do Firmware e o uso do módulo FTDI no é descrito no item 1.5.

Lolin. Ambas são equipadas com o microcontrolador ESP-12F, conhecido como NodeMCU. Uma imagem desse módulo pode ser vista na Figura 05.

Figura 04 - O módulo ESP8266 fabricado pela chinesa Ai-Thinker



Fonte: Elaborada pelo autor (2019)

Embora os módulos sejam diferentes, ambos são equipados com o mesmo microcontrolador, o ESP8266EX e possuem os mesmos blocos principais. As diferenças entre os módulos ESP-01 e ESP-12, por exemplos, estão apenas na existência ou não de um chip FTDI (para carregamento dos dados dos *firmwares* e arquivos de forma mais prática), a quantidade de pinos disponíveis para conexão. Abaixo, descrevemos algumas das portas utilizadas:

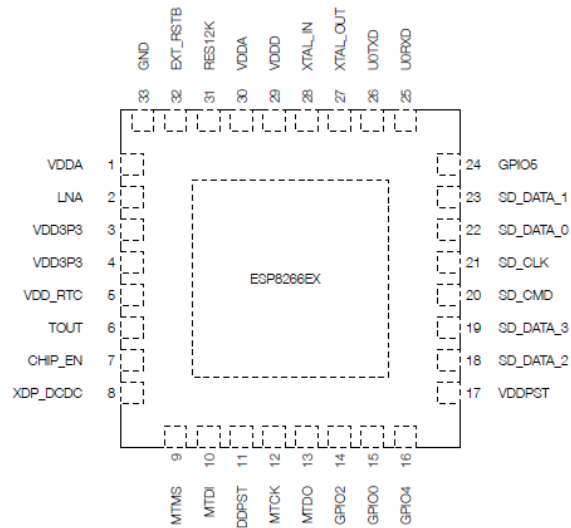
Quadro 03 – Portas utilizadas no sistema

Pino	Nome	Função
1	VDDA	Alimentação elétrica analógica de 2,5V a 3,3V
7	CHIP_EN	Ativa o microcontrolador
8	XPB_DCDC	Liga o microcontrolador após o Wakeup
14	GPIO2	UART TX durante processo de substituição do firmware e GPIO2
15	GPIO0	GPIO0 e SPI_CS2
31	RES12K	Reset
32	EXT_RSB	Sinal externo de reset
33	GND	Aterramento e alimentação elétrica negativa

Fonte: Próprio autor (2019)

Enquanto no módulo ESP-01 são apenas 04 pinos disponíveis externamente, no módulo ESP12 são 16 pinos que podem ser utilizados. O microcontrolador, no entanto é o mesmo, com 32 portas lógicas, e cada uma das portas está identificada na imagem abaixo.

Figura 05: Pinos do microcontrolador ESP8266ex.



Fonte: EXPRESSIF (2016)

Para o desenvolvimento do dispositivo, utilizou-se um microcontrolador ESP8266, uma bateria NiCd de 600mah²¹, quatro células solares de 5v por 30 mah cada ligadas em série, um diodo comum modelo 1n4007 e um sensor de temperatura e umidade DHT-22²². Para fins de identificação no texto e no código, convencionou-se chamar cada dispositivo de Node (nó), uma vez que cada um pode ser conectar a outro e também ser conectado, de maneira muito semelhante à nós atando diferentes cordas. Esse termo é comumente utilizado em tecnologia da informação para descrever dispositivos que “juntam”, “atam” comunicações, como roteadores.

²¹ Conforme testes de eficiência na tabela 02, convencionamos que para o uso em produção o ideal seria a utilização de duas baterias de 600mah, ou uma bateria de 1000mah, para uma maior autonomia do dispositivo.

²² Explicamos sobre a utilização de uma biblioteca para substituir a leitura dos dados do sensor no item 4.5.

4.2 Modelo teórico de uma rede neural baseada no módulo ESP8266

Qual a capacidade afinal de um conjunto de computadores “perceberem” o mundo? Equipamentos computacionais podem interagir, de maneira autônoma, proativa e sem intervenção humana a fim de prover alguma “inteligência” não lógica e não-mecanicista? Quais seriam os resultados dessa interação? Algumas propostas de pesquisas para esse tema seriam as possibilidades de comunicação M2M feitos por equipamentos conectados entre si através de uma rede neural.

A proposta utiliza um conjunto de módulos sem fio ESP8266 conectados onde as portas GPIO fazem o papel de Sensores, e a comunicação sem fio faz o papel das entradas e saída (no lugar dos dendritos do modelo biológico). O modelo apresenta características do modelo de McCulloch e Pitts, como ilustrado na Figura 06. Embora suas entradas na proposta teórica sejam binárias, é possível que essas entradas e as saídas assumam valores de 0 a 255, que é o limite PWM de leitura de tensão elétrica suportado pelos pinos GPIO do módulo. Também é possível que cada uma das portas seja configurada para atuar não como sensor, e sim, atuador, de modo que ao receber determinada informação um dispositivo atuaria não como dispositivo de camada de sensoriamento e passasse a executar alguma função, tal como funciona em sistemas de neurônios multicamadas²³.

Figura 06 - Modelo de neurônio artificial



Fonte: MCCULLOCH e PITTS (1943)

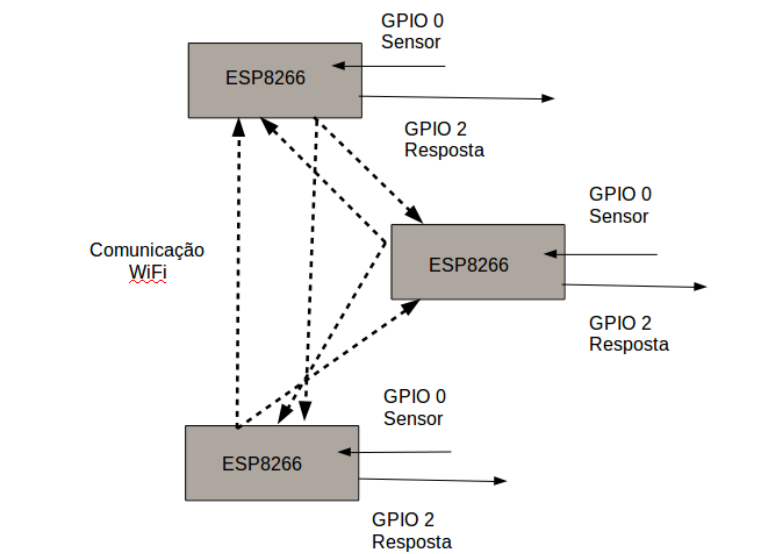
O sensor conectado ao GPIO02 será somado aos valores enviados através da rede sem fio pelos outros dispositivos ESP8266 na mesma rede. Um *script*

²³ Mencionados brevemente no capítulo 01.

carregado no *firmware* do módulo é responsável pelas funções de soma e transferência das entradas, carregando para a saída a resposta de acordo com a média das entradas dos outros módulos conectados à rede somados ao sinal recebido na entrada do próprio neurônio.

A programação do *Script* é simples e segue o seguinte raciocínio: o programa inicializa as variáveis; estabelece os valores iniciais e inicia a leitura da tensão da porta onde está conectado o sensor. Depois, inicia a conexão TCP ouvindo na porta específica; envia o valor registrado na para as conexões TCP aos outros módulos ESP8266, os nodes, sendo que a quantidade máxima de conexões é delimitada pelo hardware do dispositivo, hoje operando com no máximo de 4 dispositivos conectados simultaneamente. Para cada conexão recebida por um módulo diferente, carrega um valor em uma variável, e passa a identificá-lo de acordo com a numeração IP dentro do cabeçalho de resposta. O código tem como função principal identificar e se conectar nas redes sem fio disponíveis, que podem ser redes de infraestrutura ou um outro módulo próximo e receber os valores das conexões dos módulos e do sensor conectado à porta lógica e então realizar a somatória e média dos valores (Valor dado pela leitura da porta + Módulo01 + Módulo02 ... + MóduloN) e então compara com o limiar preestabelecido; se o valor da soma dos valores for maior que o limiar preestabelecido, envia um valor para o Gateway. A função de comparação de limiar e média podem ser alteradas a qualquer momento, por trata-se apenas de funções matemáticas simples que implementamos de acordo com o modelo estabelecido de neurônio artificial. Após isso, ele reinicia o *loop* e realiza uma nova leitura. O modelo de comunicação foi pensando para ser o mais simples possível e o mais próximo de uma rede neural biológica, como demonstrado no diagrama na Figura 07.

Figura 07 - Modelo da rede de comunicação autônoma proposta com o módulo ESP8266.



Fonte: Elaborada pelo autor (2019)

Os módulos são conectados através de uma rede sem fio, onde as portas GPIO recebem o sinal dos sensores e também atuam como saída, e a comunicação sem fio faz o papel das entradas no lugar dos dendritos do modelo biológico (Kóvacs 1996). O modelo apresenta características do modelo de McCulloch e Pitts (MCCULLOCH; PITTS, 1943) e suas entradas na proposta teórica são um número de ponto flutuante (*float*), sendo que essas entradas e saídas podem assumir valores de 0 a 255, que é o limite PWM de leitura de frequência elétrica suportado pelos pinos GPIO do módulo. Os valores recebidos nas entradas (no modelo representadas por x_n) podem ter pesos diferentes de acordo com a fórmula w_{jn} e submetidos à um processamento na função de média, que no caso do nosso modelo teórico será efetuada pelo microcontrolador do ESP8266. Em nosso modelo, o peso definido é um, mas por alteração de software esse peso pode ser alterado de acordo com os valores presentes na média. O sensor conectado ao GPIO0 será somado aos valores enviados através da rede sem fio pelos outros dispositivos ESP8266 na mesma rede.

Foi utilizado um cabeçalho HTTP para enviar e receber os dados através do node utilizando o método GET. É possível inclusive monitorar o recebimento dos dados em cada um dos nodes utilizando-se de um navegador comum de internet

através do endereço: `http://<ip_do_node>/data`, que serão disponibilizados em formato json, já antevendo a possibilidade de se armazenar esses dados e classificá-los. A escolha do HTTP como protocolo de transmissão justifica-se por simplicidade da sintaxe, por ser humanamente compreensível (*human-readable*) e por segurança²⁴, uma vez que o Micropython suporta o protocolo HTTPS que poderia prevenir a captura dos pacotes de comunicação do node na rede.

A programação do *script* é simples e apenas dois arquivos são necessários: o arquivo `boot.py` tem como função conectar o node na rede Wi-Fi disponível ou no node mais próximo, o `main.py`, que por sua vez inicia o *loop* que faz a leitura do valor dos sensores e serve requisições http (comenta aqui sobre a limitação de 1 *thread*) a cada dez segundos (o *socket* espera e trava a *thread* por 5s) e atualiza os dados nos outros nodes enviando-os através do endereço `http://<ip_do_node>/update/<node_ID>/<dados>`. Uma vez recebido, cada node faz uma média interna e atualiza seus valores de limiares. A função que realiza a leitura dos dados dos sensores pode ser vista na Figura 08.

Figura 08: Uma das funções do código

```
def sensor(next_run=None):
    if next_run and next_run > time.time():
        return None
    data = random.uniform(0, 255)
    for node, _addr in NODES.items():
        print('sending {} to node {}'.format(data, node))
        if node == ID:
            update_node_data(node, data)
            continue
        addr, port = _addr.split(':')
        path = '/update/{}/{}'.format(ID, data)
        http_client(addr, port, path)
    return time.time() + 10
```

Fonte: Próprio autor (2019)

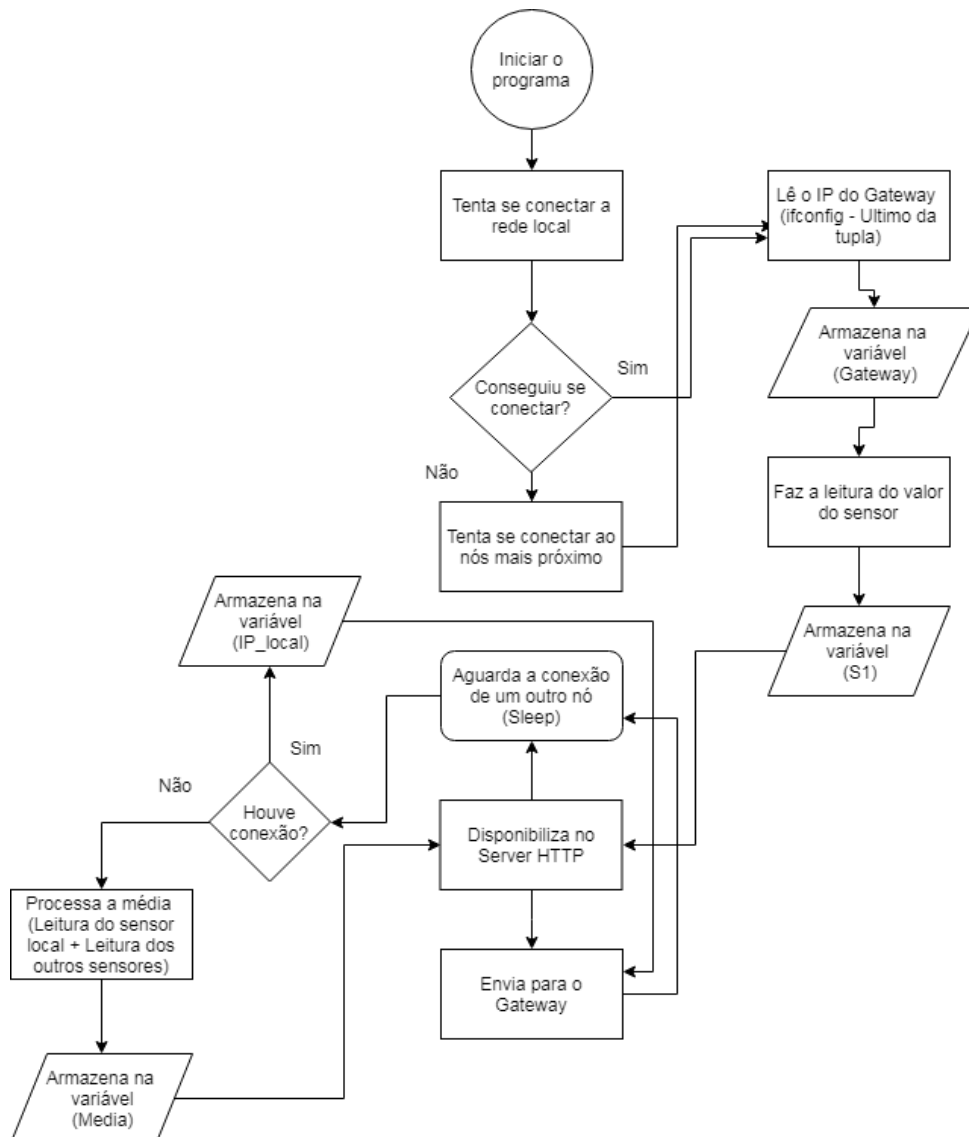
O *script* `main.py` carregado no *firmware* do módulo é responsável pelas funções de soma e transferência das entradas, carregando para a saída a resposta de acordo com a média das entradas dos outros módulos conectados à rede

²⁴ Em nosso modelo os dados trafegam de forma aberta, porém algumas camadas de segurança poderiam ser adicionadas se fossem necessárias, embora a utilização de criptografia pesada não seja possível dadas as limitações de hardware do módulo, o mesmo consegue operar em redes com criptografia WPA2, por exemplo.

somados ao sinal recebido na entrada do node. Além disso, uma vez que receba o envio de dados de um outro node, o próximo node atualiza sua tabela interna com as informações desse node, e o disponibiliza em sua tabela interna.

Portanto além da média recebida pelo dispositivo anterior, é possível saber as médias de cada um dos nodes conectados à um outro node acessando os seus dados. Um fluxograma completo do sistema pode ser observado na Figura 09.

Figura 09. Fluxograma completo de funcionamento do sistema.



Fonte: Próprio autor (2019)

4.3 Sensores, números randômicos e células solares

É importante frisar que embora se tenha realizado leituras de sensores físicos com o node²⁵, a versão final do código utiliza uma biblioteca de números randômicos para gerar os dados compartilhados na rede. Embora alguns testes tenham sido realizados utilizando-se do sensor DHT22 e um outro sensor acelerômetro GY-528 tenha sido adquirido para esse fim, durante o desenvolvimento utilizou-se principalmente uma biblioteca de números randômicos que fornecia valores no formato de ponto flutuante assim como os sensores mais comuns. Justifica-se a utilização desta biblioteca o fato de que alterações de umidade e temperatura exigem manipulação do ambiente externo, algo extremamente complexo de se simular em laboratório.

A opção de usar números randômicos permitiu um teste preciso das alterações dos valores das médias numéricas e do compartilhamento destas pelos módulos através da rede. Números randômicos são números aleatórios gerados através de algoritmos computacionais próprios, como os geradores pseudoaleatórios, que recebem uma entrada de bits aleatórios e produzem uma saída maior, como o produzido pela *RAND Corporations*²⁶, além de John Von Neuman²⁷. Em nossas simulações, utilizamos a biblioteca própria do *Python*, *Random*, para gerar esses números. Por conta disso e para facilitar o desenvolvimento do código para testes essa biblioteca fornece um valor de ponto flutuante (*float*) a cada dez segundos, conforme solicitada pelo código. De fato, essa substituição não interferiu no teste do dispositivo, uma vez que para os testes é indiferente a utilização de um valor de leitura de um sensor ou os gerados aleatoriamente. Existe também a possibilidade de utilização de outros tipos de sensores, com dados diferentes, sendo que essas alterações são possíveis com pequenas alterações no código, disponível na versão final utilizada no apêndice (Apêndice 01). Entende-se essa opção como lógica uma vez que o objeto deste

²⁵ Utilizamos o sensor DHT 22 e também leitura de dados de luminosidade com as próprias células solares que serviam de alimentação para o node.

²⁶ Vem daí o nome Random

²⁷ Um dos matemáticos pioneiro no estudo de lógica utilizada em computação e criador do computador Von Neuman, um dos primeiros computadores binários, e do método Monte Carlo

trabalho é a discussão do dispositivo e sua comunicação e não a leitura e o funcionamento dos sensores em si²⁸.

4.3.1 Células solares e baterias

Uma parte importante do projeto é a utilização de baterias recarregáveis e células solares em um sistema extremamente simples de carga, o que permite, em tese, que o sistema funcione mesmo em ambientes externos, sem necessidade de fontes ligadas à rede elétrica. Esta importante característica do projeto deve-se ao fato de que a proposta de utilização de sensoriamento em forma de troca de informações entre os nodes.

Figura 10 - Bateria de 3,6v e 600 mah



Fonte: Próprio autor (2019)

A tabela abaixo descreve os horários e o estado dos testes realizados com o conjunto de 4 células solares e uma bateria de NiCd de 600mah (seiscentos miliamperes/hora, que pode ser vista na Figura 10) durante dois dias, em um ambiente aberto.

²⁸ Futuramente pode ser explorado em trabalhos posteriores.

Tabela 01 - Medição de tensão do sistema de Bateria NiCd/Célula Solar

Data	Hora	Descrição do tempo/Medição da tensão da célula	Tensão da bateria(V)
03/11	17:15 ²⁹	Nublado/Célula a 4,32v	4,02
03/11	18:15	Poente/Célula a 2,55v	3,99
03/11	19:15	Luz branca artificial/Célula a 0,7v	3,96
03/11	21:10	Noite	3,94
03/11	22:15	Noite	3,93
04/11	00:40	Noite	3,91
04/11	09:00 ³⁰	Sol/Célula a 3,00v	3,08
04/11	13:00	Sol/Célula a 5,30v	4,15
04/11	16:00	Sol/4,86v	4,15
04/11	17:50	Sol/Célula a 4,06	4,06
04/11	22:20	Noite	4,00
05/11	20:00	Noite	3,9

Fonte: Próprio autor (2019)

A leitura dos valores da tabela observados no experimento demonstra que um conjunto de quatro células de medidas de 68 por 37mm modelo XY68X37 foram mais do que suficientes para a carga de uma bateria. Bem como a utilização do diodo na ligação do sistema de carga permite que a bateria se mantenha carregada mesmo durante a noite. A célula solar utilizada em nosso experimento oferece tensão de 5v e 60mah quando exposta ao sol, e pode ser observada em detalhes na Figura 11. Na versão final do protótipo teve-se o cuidado de fazer as ligações por

²⁹ Horário normal.

³⁰ Horário brasileiro de verão.

conectores externos e de forma modular, permitindo assim posteriormente a utilização de mais ou menos módulos de energia solar de acordo com a potência da bateria, luminosidade do ambiente ou consumo do dispositivo, que pode oscilar com a implementação futura de bibliotecas específicas de consumo conforme foi descrito no item 4.4.

Figura 11 – Célula solar de 5v e 60 mah.



Fonte: Próprio autor (2019)

Tão ou mais importante do que os componentes de hardware acima descritos estão os componentes de software, disponíveis no próximo item. De fato, a integração de hardware e software compatível apresentou-se como um grande desafio para o desenvolvimento do dispositivo, e apesar disso, a variedade de bibliotecas disponíveis na linguagem Micropython foi essencial para o desenvolvimento deste.

4.4 Bibliotecas

A linguagem Micropython guarda como característica principal um conjunto de bibliotecas otimizadas para o uso com microcontroladores e outras placas embarcadas. De acordo com a documentação oficial do projeto, as configurações mínimas para uma placa embarcada rodar uma aplicação Micropython é sugerida com um processador de pelo menos 128KB de ROM e no mínimo 8 KB de memória

RAM³¹. Os criadores dizem ainda que determinadas funções, como o terminal nativo, podem ser desabilitadas para uma performance melhor. Nosso microcontrolador atende com folga essas especificações, uma vez que possui 1MB de EPROM e 64KB de memória RAM

Deste modo, foram encontradas algumas bibliotecas e módulos que têm como função principal a utilização com equipamentos cujas características estão de acordo com nossa descrição de dispositivos para IoT. Entre eles, o que demanda atenção deste trabalho são funções dedicadas ao uso eficiente e economia de energia elétrica e a possibilidade de se trabalhar com sinais discretos de sensores.

O módulo *machine* tem suas funções voltadas para o controle do hardware, normalmente com acesso direto à informações e alterações em funções como velocidade da CPU, *timers*, controle de pinos e outros. De fato, a utilização de tais bibliotecas demanda cautela, uma vez que utilizadas incorretamente podem gerar danos ao dispositivo, como mal funcionamento, resets inesperados ou mesmo danos definitivos ao hardware (GEORGE, 2014).

Com a proposta de utilização do módulo ESP8266 como um sensor nó completo, com microcontrolador, bateria e interface de comunicação wireless (Roy et al., 2018), é necessário que o mesmo seja capaz de alternar entre um modo de consumo de energia, para realizar a leitura de um sensor por exemplo, e voltar ao modo de ‘hibernação’, uma vez que esses sensores devem funcionar pela maior quantidade de tempo possível recebendo e enviando informações antes de ser necessária a substituição da bateria, no entanto a função que permite a reativação do sistema através da leitura do sensor ainda não foi implementada na versão utilizada em neste sistema³².

Algumas das bibliotecas que compõem o módulo *machine*³³ têm seu funcionamento voltado justamente para esse processo ‘*awake-sleep*’. A começar pela função IRQ, que tem por função habilitar e desabilitar interrupções. Em aplicações de baixo nível é comum existirem funções como esta, que interrompem

³¹ O tamanho mínimo da ROM é de 80K, e embora seja possível inicializar o sistema com uma memória RAM de 4MB, não seria possível realizar muita coisa, exceto interpretar algumas instruções simples

³² Uma função de callback da função *machine.sleep*;

³³ Em nosso sistema, utilizamos a versão 1.93 do firmware Micropython

um determinado fluxo com a alteração de um sinal externo (como é o caso da interrupção do processamento de um thread, ou um fluxo, após digitar em uma tecla do teclado em um computador). O módulo *machine* dispõe de opções onde uma leitura de sinal elétrico em um de seus pinos gera uma requisição de interrupção (IRQ), de maneira que configurando um dos pinos para uma leitura de um sensor de luminosidade, por exemplo, ao detectar uma mudança nesse pino específico enviaria um sinal ao dispositivo ESP8266, que sairia do estado de hibernação³⁴. Várias outras bibliotecas foram essenciais no desenvolvimento do dispositivo. Abaixo foram listadas as principais e quais de suas funções foram utilizadas.

Quadro 04 - Bibliotecas utilizadas.

Biblioteca	Descrição	Funções utilizadas
machine	Contém funções especificamente relacionadas ao hardware de uma determinada placa	Função pin na leitura dos dados dos sensores
network	Fornecer drivers de rede e configuração de roteamento além das interfaces	Utilizada para disponibilizar a rede sem fio local no modo AP, conectar-se à outra rede no modo Station e também identificar os endereços de IP e gateway em cada um dos nodes
socket	Função de abertura de sockets na rede	Função de disponibilizar o servidor HTTP e conectar aos demais servidores disponibilizados
time	Função interna de contador em milissegundos.	Para contagem de tempo nas operações do sistema

Fonte: Adaptado de GEORGE et al. (2017)

³⁴ A função *modem-sleep* permite ser utilizada nesses casos, porém sua implementação exige a existência de um sinal elétrico do tipo PWM em um dos pinos, ou seja, o dispositivo necessariamente deve estar conectado à um sensor capaz de gerar sinal externo

A biblioteca Network é carregada já durante o *boot* do sistema, e fica apropriadamente armazenada no *script* 'boot.py' cuja função é tentar realizar a conexão do node com a internet (através de uma rede WiFi externa já pré-definida no código) e caso não seja possível, realizar a conexão ao módulo mais próximo assim como disponibilizar uma conexão sem fio para que outros nodes se conectem nela. Essa biblioteca, conforme a descrição no quadro, é a responsável por estabelecer o protocolo TCP/IP e endereçamento de IP versão 4, configurando o endereço de IP do node e sua máscara de rede. É através dela também que o node é capaz de identificar os endereços de rede local e dos outros nodes à ele conectado, seja no modo de estação (STA), ou seja, o node que disponibilizou à rede no qual o módulo se conectou, seja no modo de ponto de acesso (AP), que seriam outros módulos conectados na rede por ele disponibilizados. Todos esses dados são obtidos através de uma função dessa biblioteca que retorna esses dados, sendo então carregados na variável e disponibilizados no código, conforme descrito anteriormente.

5 JUSTIFICATIVA

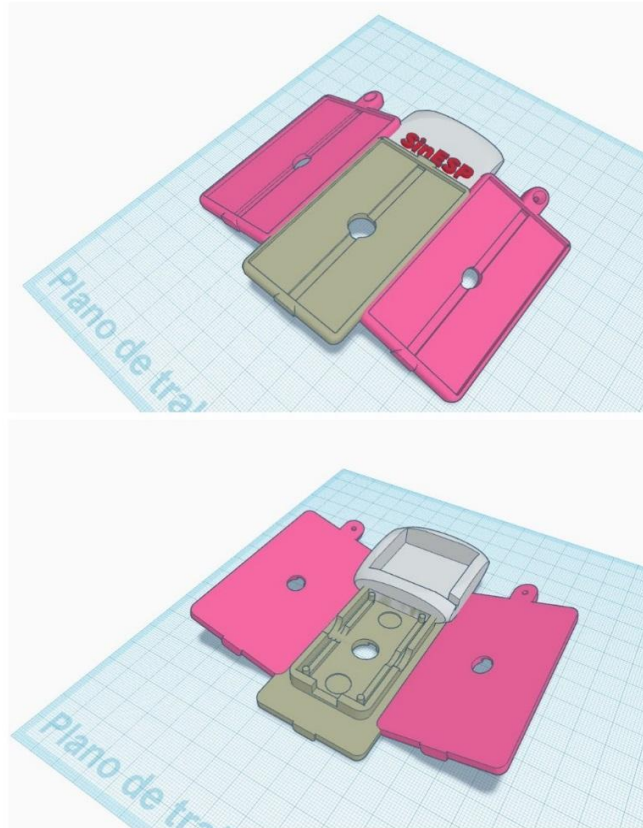
A escolha desta proposta de modelo de neurônio artificial baseou-se nas características do dispositivo de poder conectar-se simultaneamente a uma rede sem fio ao como uma estação ao mesmo tempo em que tem outros dispositivos conectados à ele, uma característica comum em alguns dispositivos de rede, como os roteadores, que são os nós responsáveis por encaminhar os dados através de diferentes redes na internet. De fato, em uma analogia com os roteadores, esses nodes receberam informações, as armazena e ao receber sinais de seu próprio sensor, realizam uma média e enviam esses dados para o próximo nó que estiver conectado. A possibilidade de um mesmo sinal ser 'percebido' por mais de um nó separado fisicamente de outro, assim como ocorre em um sistema nervoso, permitiria uma leitura mais 'humana' de dados, e não tão digital.

Assim como se percebem as nuances de cores e cheiros, uma rede de dispositivos espalhados 'sentindo' a leitura de outros sensores poderiam sentir as 'nuances' das alterações desses dados, quando fossem mais intensas ou mais lentas, por exemplo. Além disso, utilizando sensores físicos, a distância física entre eles provocaria mudanças, mesmo que sutis, nos dados das leituras. Também existe a possibilidades de se utilizarem nodes sem a etapa de sensores, onde o sinal de um sensor seria apenas repassado para o próximo node, sem interação, de forma análoga ao que acontece no sistema nervoso humano, onde alguns nervos têm funções de sensibilidade e outros apenas de transmissão de informação.

É importante frisar que 'média' aqui é utilizada no sentido de operação matemática. De fato, pode-se realizar uma média dos dados, dividindo-se o valor dos dados obtidos a partir de dois nodes (ou dois neurônios, por exemplo) por dois, porém, ao se enviar esses dados para o próximo node, esse dado passa a ser apenas um. A função do sistema é, portanto, não estabelecer uma média matemática geral dos nodes, e sim, uma percepção geral do estado dos sensores conectados à rede. Não se trata, portanto, de uma rede de precisão matemática, e sim, de sensoriamento compartilhado. Justifica também essa proposta o fato de se utilizarem tecnologias modernas como carregamento das baterias baseado em células solares, linguagem de programação de código aberto e tecnologias de ponta

como prototipagem em 3D, como pode ser vista na imagem de um dos protótipos³⁵ enviados para impressão 3D, na Figura 12.³⁶

Figura 12 – Imagem de um protótipo enviado para impressão em 3D.



Fonte: Próprio autor (2019)

³⁵ Observe que ao contrário dos testes da seção 4.3.1, optamos por uma caixa para o protótipo com apenas três células solares ao invés de quatro, uma vez que em nossa proposta já prevemos um uso de energia menor, dada a possibilidade de interrupção com uso de sensores físicos.

³⁶ Embora tenhamos utilizado a impressão 3D no projeto, ela não faz parte da abordagem do dispositivo. Apenas foi a maneira mais fácil encontrada para a prototipagem do dispositivo para uso externo durante os testes e para apresentação

6 RESULTADOS OBTIDOS

Ao acessar os endereços IP de cada um dos nodes conectados na rede através do endereço `http://<ip_do_node>/data` deve ser possível visualizar os dados do próprio módulo, a média dos valores recebidos dos outros módulos e os valores enviados por cada um dos módulos, conforme mostra a Figura 13. É importante frisar que o *script* desenvolvido não trabalha com sincronicidade, o que significa que os dados podem ser recebidos dentro da janela de tempo dos 05 segundos previstos no código. Por conta disso, essa média não pode ser utilizada por aplicações de precisão, mas pode entregar informações sobre os valores apurados pelos sensores de forma distribuída e autônoma, uma vez que as informações podem ser acessadas localmente em cada módulo sem a necessidade que os outros estejam conectados ou em funcionamento.

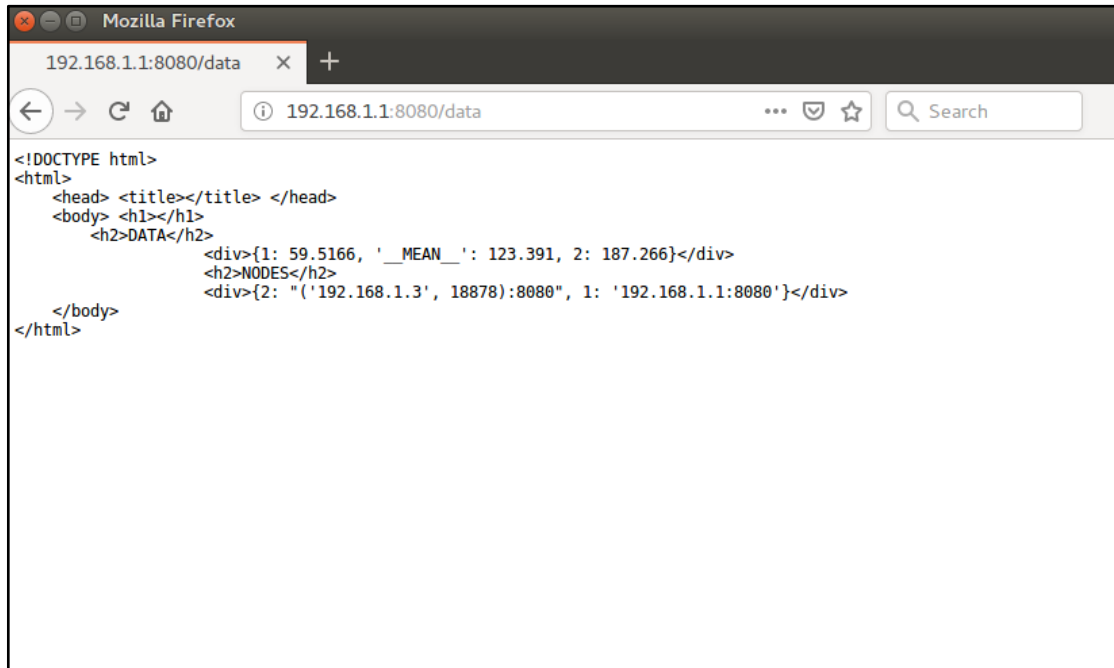
Figura 13 – Visualização dos dados no formato recebido pelo navegador

```
2: "( '192.168.1.4', 13820):8080", 3: "( '192.168.1.2', 3307):8080"
```

Fonte: Próprio autor (2018)

Na Figura 14, mostra-se uma tela de um navegador comum de internet conectado a um node, que recebe em tempo real as informações de outros dois nodes, e partir dos dados deles, gera sua própria média e a enviar para o próximo node. Lembrando que esse próximo node pode receber os dados e tratá-lo, se for um servidor conectado à internet, por exemplo. O primeiro valor, 1: 59,5166, foi gerado pelo gerador de número randômicos e cumpre a função de uma leitura feita pelo sensor. O segundo, MEAN, é a média do valor obtido pelo sensor do node que se está acessando pelo navegador (nesse caso, o node de ID 1) e o valor recebido pelo segundo sensor, que foi enviado pela rede sem fio por um outro sensor conectado a esse através da rede sem fio. Ao observar-se a Figura 15, encontram-se os dados obtidos na consulta ao node através do navegador.

Figura 14 – Navegador de internet Mozilla acessando os dados do node



Fonte: Próprio autor (2019)

Um node que estiver realizando um processo de transmissão recebe esses dados, adiciona aos seus e os envia para o próximo, conforme se pode ver na figura 15, obtida através da leitura em tempo real dos dados de dois nodes conectados através de uma porta serial a um computador³⁷. Ao lado direito tem-se a leitura da saída do terminal realizada pelo node de ID 2, que está conectado no modo Estação (STA) ao node de ID 1, e após realizar a leitura interna do seu sensor, atualiza sua média e a encaminha para o gateway da rede, no caso o Node de ID 1. Se o Node de ID estivesse conectado à internet, esses dados seriam recebidos por um servidor, que os trataria de acordo com o seu número de IP. É importante frisar que não existe comunicação intermediada pelo computador, que no exemplo abaixo somente faz a leitura do terminal, de modo que toda comunicação se dá única e exclusivamente através da rede sem fio, em ambos sentidos.

³⁷ Embora os nodes estejam conectados ao computador, sua comunicação se dá de forma independente, através da rede sem fio. Posteriormente esses dados poderiam ser informados através de um display TFT, por exemplo.

Figura 15 – Processo de transmissão dos dados através dos nodes

```

Terminal
mira@Mira-Asus: ~/Node - 2018/Casa
<!DOCTYPE html>
<html>
  <head> <title></title> </head>
  <body> <h1></h1>
  updated: node=2 value=119.531
</body>
</html>
[connection w/ ('192.168.1.3', 17624) closed]
ERROR: cannot receive connection from ('192.168.1.3', 17624): [Errno 110] E
TIMEDOUT
[connection w/ ('192.168.1.3', 18878) started]
serving:
method=GET path=/update/2/187.266
response:
updating locally node 2: 187.266
calculating mean...
new node mean: 123.391
<!DOCTYPE html>
<html>
  <head> <title></title> </head>
  <body> <h1></h1>
  updated: node=2 value=187.266
</body>
</html>
[connection w/ ('192.168.1.3', 18878) closed]
[connection w/ ('192.168.1.4', 55534) started]
serving:
method=GET path=/data
response:
<!DOCTYPE html>
<html>
  <head> <title></title> </head>
  <body> <h1></h1>
  <h2>DATA</h2>
  =223.125\n </body>\n</html>'
ERROR: cannot receive connection from ('192.168.2.1', 8080): [Errno 110] E
TIMEDOUT
sending 239.063 to node 2
updating locally node 2: 239.063
calculating mean...
new node mean: 239.063
sending 239.063 to node 1
DEBUG: response from 192.168.1.1: b'<!DOCTYPE html>\n<html>\n  <head> <t
  title></title> </head>\n  <body> <h1></h1>\n    updated: node=2 value
  =239.063\n </body>\n</html>'
ERROR: cannot receive connection from ('192.168.2.1', 8080): [Errno 110] E
TIMEDOUT
sending 119.531 to node 2
updating locally node 2: 119.531
calculating mean...
new node mean: 119.531
sending 119.531 to node 1
DEBUG: response from 192.168.1.1: b'<!DOCTYPE html>\n<html>\n  <head> <t
  title></title> </head>\n  <body> <h1></h1>\n    updated: node=2 value
  =119.531\n </body>\n</html>'
ERROR: cannot receive connection from ('192.168.2.1', 8080): [Errno 110] E
TIMEDOUT
sending 187.266 to node 2
updating locally node 2: 187.266
calculating mean...
new node mean: 187.266
sending 187.266 to node 1
DEBUG: response from 192.168.1.1: b'<!DOCTYPE html>\n<html>\n  <head> <t
  title></title> </head>\n  <body> <h1></h1>\n    updated: node=2 value
  =187.266\n </body>\n</html>'
ERROR: cannot receive connection from ('192.168.2.1', 8080): [Errno 110] E
TIMEDOUT
sending 221.133 to node 2
updating locally node 2: 221.133
calculating mean...
new node mean: 221.133
sending 221.133 to node 1

```

Fonte: Próprio autor (2019)

Pode-se concluir através do que foi demonstrado neste artigo que um modelo de módulos autônomos se comunicando diretamente sem intervenção humana é viável, sendo que uma rede neural autônoma atuando com monitoramento e resposta através de uma rede sem fios poderia ser utilizada amplamente em aplicações futuras voltadas à análise de padrões de comunicação de redes M2M, além de possibilidades de interações entre máquinas e equipamentos, encaminhando informações referentes a essas mudanças de estados em um repositório comum, por exemplo.

Em ambos os casos, esses domínios são de amplo interesse da computação ubíqua e da IoT. Além disso, durante a pesquisa foram encontradas outras características importantíssimas para o domínio da comunicação M2M. A alimentação do sistema utilizando-se de um pequeno conjunto de placas solares sem dúvida foi essencial para a proposta de um projeto autônomo. Nos testes, foi possível a obtenção de cargas completas da bateria em dias de sol forte entre 11 de

novembro e 12 de dezembro com a utilização de um conjunto de três placas solares nos horários entre 10:00h e 14:00h; de fato, o dispositivo pode ser utilizado por um longo período de tempo sem necessidade de recarga desde que seja respeitado seu consumo elétrico de acordo com a potência da bateria. Nestes testes foi utilizada uma bateria de 600ma, conforme descrito no item 4.1, porém a utilização de bibliotecas de consumo de energia permitirá, em aplicações futuras, a utilização do módulo por alguns dias sem necessidade de recarga³⁸. Na Figura 16 pode-se ver a versão do dispositivo utilizado nos testes, com três células solares, bateria de 600mah e caixa de contenção prototipada com uma impressora 3D.

Figura 16. Dispositivo modular operando com três células de 5v e 60mah



Fonte: Próprio autor (2019)

A abordagem proposta para implementar um sistema de IoT para criação de valor sustentável consiste em alavancar o conhecimento de domínio / especialista de um produto, processo e / ou sistema por meio de projeto paralelo do hardware do sistema e da arquitetura de IoT. Além do elemento de design, o outro componente essencial para a abordagem proposta é combinar o domínio / conhecimento

³⁸ Tempo estimado considerando um consumo de 10mah com picos de 500mah por um minuto a cada 30, com uma bateria de 600mah, no modo *deep.sleep*.

especializado com um algoritmo de aprendizado de máquina. Este algoritmo de aprendizado de máquina permite que mais informações sejam extraídas da rede de sensores IoT do que seria tradicionalmente medida (Bradley et al., 2017).

7 OUTROS RESULTADOS OBTIDOS

7.1 Apresentação EBICC

Uma versão preliminar deste projeto foi exibida durante o Encontro Brasileiro Internacional de Ciência Cognitiva (EBICC) em 2017, na USP de São Paulo. Na ocasião, apresentou-se a proposta e as possibilidades de utilização do dispositivo que ainda estava em fase de testes. Houve interesse inclusive do Professor Walter Teixeira Lima Júnior, Professor Doutor da Universidade Federal do Amapá, autor de um projeto sobre um sistema que tornasse possível registrar alterações em ambientes da floresta amazônica.

Figura 17. Apresentação de Artigo no EBICC 2017.



Fonte: Próprio autor (2017)

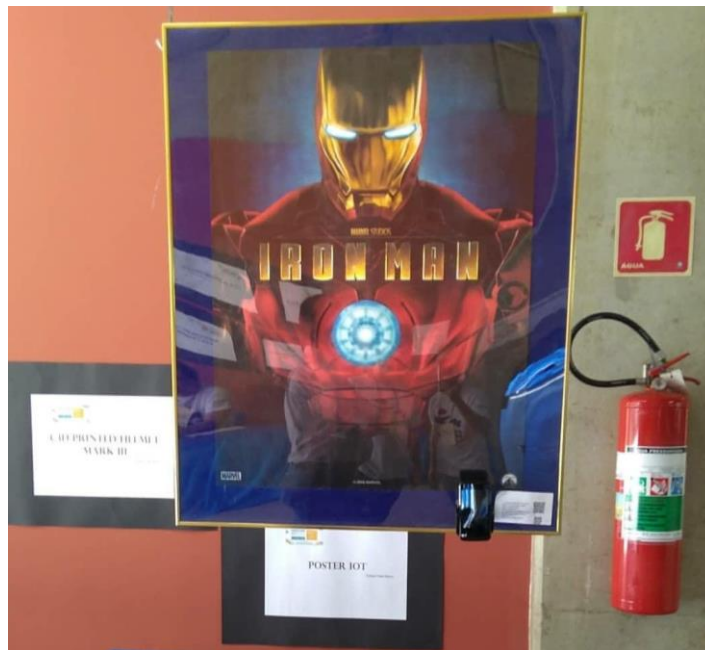
7.2 Poster IoT

Outras possibilidades de utilização do microcontrolador ESP8266 foram apresentadas, como por exemplo no evento Arduino Day, um evento internacional

organizado localmente na Fatec Bauru, onde são exibidos projetos ligados à cultura maker, automação e IoT.

A apresentação contou com um pôster de filme, onde um microcontrolador ESP8266 rodando uma programação em Micropython permitia que os visitantes controlassem a ativação da iluminação dos olhos e do peito do personagem, atrás de LEDs instalados na parte traseira do pôster. Além disso, o microcontrolador estava conectado em tempo real à internet, permitindo que informações sobre a interação fosse enviada em tempo real para a internet, por exemplo.

Figura 18. Apresentação de Pôster com tecnologia IOT do ESP8266 no Arduino Day da Fatec Bauru.



Fonte: Próprio autor (2018)

8 PARCERIAS INSTITUCIONAIS

Foi de extrema importância a parceria da Faculdade de Tecnologia (Fatec) de Bauru, que disponibilizou espaço para a demonstração de etapas do trabalho, como a feira mencionada acima, bem como o trabalho de diversas pessoas, como o Professor Rafael Balan, que prestou assessoria na impressão 3D da caixa de

contenção do dispositivo, do Professor Me. Adriano Mazotti, Professor Me. Claudines Torres, Prof. Me. Alexandre Galvani, Prof. Dr. Tadeu Pellison e Professor Paulo Sérgio Pereira, com os quais se compartilhou experiências e informações relativas aos dispositivos microcontroladores para utilização em aplicações de IoT utilizados no projeto assim como outros que não foram utilizados. O auxiliar docente Cleiton Carvalho ajudou a elaboração da proposta de alimentação de energia solar, fornecendo *know-how* e compartilhando conhecimentos sobre a utilização de equipamentos e técnicas de eletrônica. Ademais, houve apoio irrestrito dos setores administrativos durante todo o período de desenvolvimento e testes da proposta, graças ao comprometimento da Direção da Unidade e das Diretorias administrativas com iniciativas de desenvolvimento tecnológico como a descrita nesse relatório.

9 IMPACTOS

Entre os principais impactos que se pode destacar no desenvolvimento do projeto, os principais são relacionados às propostas e as tecnologias empregadas no processo de desenvolvimento do dispositivo.

9.1 Dispositivo autônomo

Desenvolver um dispositivo microcontrolador autônomo com a utilização de células solares permite que o mesmo tenha maior autonomia de uso em ambientes externos, graças à possibilidade de se dispensar sistemas de carga e de alimentação elétrica. A possibilidade de se conectarem nodes em sequência, um enviando dados para o próximo fisicamente distante também possibilita a utilização desses dispositivos para uma arquitetura de rede própria, com uma comunicação M2M independente da internet.

9.2 Prototipagem em 3D

Criar uma caixa para armazenamento e proteção do microcontrolador bem como para acomodação da bateria e das células solares foi uma tarefa relativamente simples graças à utilização de ferramentas de criação e impressão 3D. O site <http://www.tinkercad.com>, criado e mantido pela Autodesk® foi utilizado para a criação do modelo em formato STL que depois foi enviado para impressão em uma impressora 3D do tipo Stella, construída por um dos colaboradores da Fatec Bauru.

9.3 Segurança da informação

Uma das preocupações durante o desenvolvimento do projeto esteve relacionada a futuras implicações relacionadas à segurança dos dados. Embora não trabalhemos com dados seguros no dispositivo desenvolvido, a implementação da comunicação baseada no protocolo HTTP permite que esses dados sejam enviados através de um protocolo do tipo HTTPS, uma versão do protocolo que trabalha com chaves criptográficas públicas e privadas, o que permitiria que apenas os nodes que possuísem a chave pudesse enviar e receber os dados. Dispositivos que se conectassem ao sistema poderiam “ouvir” a comunicação, mas não seriam capazes de interpretá-la.

10 DIFICULDADES

10.1 Importação e documentação, hardware fechado

Embora alguns desses microcontroladores sejam encontrados no Brasil, o custo de importação acaba tornando mais viável a compra direta através de distribuidores chineses. Em sites como o Aliexpress, um *marketplace* chinês, é possível encontrar modelos do ESP12 a partir de US\$ 3,00 (três dólares), e em versões simplificadas a partir de US\$ 2,00 (dois dólares). O custo de importação fica entre R\$ 12,00 e R\$15,00 e quando taxados pelos correios, a alíquota fixa de importação é de apenas R\$ 15,00, o que compensa a compra de tais microcontroladores em certa quantidade, uma vez que no Brasil eles são

encontrados a partir de R\$ 30,00, cada. A demora e a incerteza da entrega por parte dos correios, no entanto, pode ser um complicador³⁹.

Uma das maiores dificuldades, no entanto, foi com questões relacionadas à documentação do software e do hardware. Alguns fabricantes chineses, como a Wemos.cc não disponibilizava um manual completo da sua versão da placa ESP8266, e embora o *datasheet* do microcontrolador distribuído pelo AI-Thinker seja completo, a falta de um manual especializado para a placa causou percalços como a queima de componentes durante os testes com bateria. Apesar disso, o maior problema sem dúvida foi relacionado à documentação do *firmware* Micropython. Havia bibliotecas documentadas que não eram funcionais, como por exemplo uma das saídas que permitiria o Wake-up do deep-sleep do dispositivo⁴⁰. Embora consideremos que esses sejam problemas superficiais, trabalhar com documentação incorreta aumenta os erros e conseqüentemente o tempo durante a fase de produção.

³⁹ Valores de Novembro de 2018.

⁴⁰ Tais problemas foram reportados aos desenvolvedores

11 CONSIDERAÇÕES FINAIS

De acordo com os testes realizados antes e durante o desenvolvimento do produto, determinou-se que o desenvolvimento de uma neurônio artificial, ou um dispositivo de comportamento análogo ao neurônio com a utilização de um microcontrolador sem fio e de uma linguagem de programação aberta é não só possível, como viável e de baixo custo, permitindo que uma grande rede desses pudesse ser criada para envio e compartilhamento de informações em tempo real mesmo que não houvesse presença de energia elétrica ou internet no local.

Além disso, ao demonstrar o funcionamento de um sistema de funcionamento análogo ao neurônio humano com sistemas de baixo custo, apesar das limitações do hardware do microcontrolador utilizado, pôde-se demonstrar as possibilidades criadas com o advento da comunicação M2M no âmbito da IoT e como isso pode impactar futuramente o funcionamento e a implementação de tecnologias ubíquas e pervasivas, de modo que a criação de um ambiente sensível onde a interação com a tecnologia da informação possa se dar de maneira invisível e não consciente se torna viável.

REFERÊNCIAS

- AI THINKER. 2015. **ESP-01 Wi-Fi module**. Versão 1.0. 2015. Disponível em: <<https://ecksteinimg.de/Datasheet/Ai-thinker%20ESP-01%20EN.pdf>>. Acesso em: Out 2017.
- AMAZONAS, J. A internet das Coisas veio para ficar. **Revista Fonte**, ano 10, n. 13, 2013.
- ANISH, R.; KIRUBAKARAN, S. Design of Embedded Agent for Systems Based On Internet of Things. **International Journal of Advanced Research in Electronics and Communication Engineering (IJARECE)**, v. 5, n. 4, p. 843-845, 2016. Disponível em : <<http://ijarece.org/wp-content/uploads/2016/04/IJARECE-VOL-5-ISSUE-4-843-845.pdf>>. Acesso em: Abr 2018.
- ATZORI, L. et al. The Internet of Thing: A survey. **Computer Networks Elsevier**, 2010. Disponível em: <<https://www.sciencedirect.com/science/article/pii/S1389128610001568>>. Acesso em: Out 2017.
- BABER, C.; BAUMMAN, K. Embedded human computer interaction. **Applied Ergonomics**, v. 33, n. 3, p. 273-287, 2002.
- BRADLEY et al. Parallel Design of a Product and Internet of Things (IoT) Architecture to Minimize the Cost of Utilizing Big Data (BD) for Sustainable Value Creation. **The 24th CIRP Conference on Life Cycle Engineering**, v. 61, p. 58 – 62, 2017. Disponível em: <<https://www.sciencedirect.com/science/article/pii/S2212827116313828>>. Acesso em: Fev 2018.
- DICOLA T. 2014. **Adafruit MicroPython Tool**. Disponível em: <<https://github.com/adafruit/ampy>>. Acesso em: Jun 2017.
- ELECTRODRAGON. 2017. **ESP8266 ESP12-F DATASHEET**. Disponível em: <HTTP://WWW.Electrodragon.com/w/ESP-12F_ESP8266_Wifi_Board>. Acesso em: Abr 2018.

ESPRESSIF SYSTEM. 2015. **ESP8266EX Datasheet. Version 4.3**. Disponível em: <espressif.com>. Acesso em: Mai 2018.

GEORGE D. P. et al. 2014. **MicroPython Documentation**. Disponível em: <<https://micropython.org/>>. Acesso em: Jun 2017.

GOUVEIA, P. R. N. T. Convergência de redes sem fios para Comunicação M2M e Internet das Coisas em Ambientes Inteligentes. **Universidade da Beira Interior**, 2013. Disponível em: <https://ubithesis.ubi.pt/bitstream/10400.6/1892/1/Disserta%C3%A7%C3%A3o_Mestrado_Paulo_T_Gouveia.pdf>. Acesso em: Set 2017.

GRATTON, A. 2015. **ESP82566 and ESP82 serial bootloader Utility**. Disponível em: <<https://github.com/espressif/esptool>>. Acesso em: Jun 2017.

HUANG, J. et al. Optimizing M2M communications and quality of services in the IoT for sustainable smart cities. **IEEE Transactions on Sustainable Computing**, p. 1-12, 2017.

IEEE. IEEE Standard Glossary of Software Engineering Terminology. **IEEE Std 610.12-1990**, p. 1–84, 1990. Disponível em: <<https://ieeexplore.ieee.org/document/159342>> Acesso em: Jun 2017.

JACOBSSON, A.; DAVIDSSON, P. Towards a model of privacy and security for smart homes. **Internet of Things (WF-IoT), 2015 IEEE 2nd World Forum on. IEEE**, p. 727-732, 2015.

JOHNSON-LAIRD, P. N. Mental models and reasoning. **Leighton, J.P. e Sternberg, R.J. (Eds.), The Nature Of Reasoning**, p. 169-204, 2004.

KÓVACS, Z. L. **Redes neurais artificiais**. Collegium cognition: São Paulo, 1996.

NODEMCU COMPANY. 2017. **NodeMCU Documentation**. Disponível em: <<https://nodemcu.readthedocs.io/en/master/>>. Acesso em: Mar 2018.

MIRA, J. E. et al. Proposta de rede de comunicação M2M autônoma com Micropython e o módulo wi-fi esp8266. In: **XI International Brazillian Meeting on Cognitive Science**, Caderno de Resumos. São Paulo, 2017. Disponível em: <http://ebicc.sbccc.org.br/pdf/abstract_book.pdf>. Acesso em: Abr 2018.

MCROBERTS, M. **Arduino básico**. São Paulo: Novatec Editora. 2011.

MCCULLOCH W., PITTS, W. A Logical calculus of the ideas immanent in nervous Activity. **Bulletin of Mathematical Biophysics**, v. 5, p. 115-133, 1943. Disponível em: <<http://www.cse.chalmers.se/~coquand/AUTOMATA/mcp.pdf>>. Acesso em: Mai 2016.

PEREIRA, S. L. 2008. **Introdução a Inteligência Artificial**. Disponível em: <<http://www.ime.usp.br/~slago/>>. Acesso em: Abr 2016.

PERERA, C. Fog computing for sustainable smart cities: A survey. **ACM Comput. Surv.**, v. 50, n. 3, p 32-43, 2017.

PINOCHET, L. **Tecnologia da informação e comunicação**. - Rio de Janeiro: Elsevier, 2014.

RICHARDSON, M.; WALLACE, S. **Getting Started With Raspberry Pi**. Maker Media, 2016. Disponível em: <<https://books.google.com.br/books?id=xYhMliITwC4C&lpq=PR2&ots=W45kIDjw2&dq=raspberry%20pi%20description&lr&hl=pt-BR&pg=PR2#v=onepage&q=raspberry%20pi%20description&f=false>>. Acesso em: Mai 2018.

ROY, S. S. et al. Building a Sustainable Internet of Things: Energy-Efficient Routing Using Low-Power Sensors Will Meet the Need. In: **IEEE Consumer Electronics Magazine**, v. 7, n. 2, p 42-49, 2018.

RUSSEL, Stuart J. (Stuart Jonathan), 1962- **Inteligência artificial**. Stuart Russell, Peter Norvig; tradução Regina Célia Simille. – Rio de Janeiro: Elsevier, 2013.

SANTAELLA, L. Mídias Locativas: A Internet Móvel de Pessoas e Coisas. **Revista FAMECOS**, p. 96-102, 2008.

SACCOL, A. Z.; REINHARD, N. Tecnologias de informação móveis, sem fio e ubíquas: definições, estado-da-arte e oportunidades de pesquisa. **Rev. adm. contemp.**, Curitiba , v. 11, n. 4, p. 175-198, 2007.

SCOLARI, S. H. P. ; MARAR, J. F. A natureza das emoções em produtos: estudos preliminares. **Educação Gráfica (UNESP - Bauru)**, v. 13, p. 205, 2009.

SINGER, T. Tudo conectado: conceitos e representações da internet das coisas. In: **SIMSOCIAL: simpósio em tecnologias digitais e sociabilidade 02**, UFBA Bahia. 2012. Disponível em: <<http://gitsufba.net/anais/simsocial-2012>>. Acesso em: Mar 2016.

TEIXEIRA, J. F. 2016. **Filosofia da Mente – a ética da singularidade**. Disponível em: <<https://www.akimneto.com.br/index.php/artigos/detalhes/filosofia-da-mente-a-etica-da-singularidade/64>>. Acesso em: Jun 2017.

Turing, A. M. 1950. **Computing Machinery and Intelligence**. Mind 49: 433-460. Disponível em: <<https://www.csee.umbc.edu/courses/471/papers/turing.pdf>>. Acesso em: Fev 2019.

WEMOS. 2018. **Wemos wiki**. Disponível em: <<https://wiki.wemos.cc/doku.php>>. Acesso em: Set 2018.

WEISER, M. The computer for the 21st Century. **Scientific American**, v. 265, p 94-104, 1991.

WEINBERG, B. D. et al. Internet of Things: Convenience vs. privacy and secrecy. **Business Horizons**, v. 58, n. 6, p. 615-624, 2015.

ZILIO, D. Inteligência artificial e pensamento: redefinindo os parâmetros da questão primordial de Turing. **Ciência cognitiva**, Rio de Janeiro, v. 14, n. 1, p. 208-218, 2009.

APÊNDICES

APÊNDICE 01- Código do Softwares SinESP 1.0

Arquivo boot.py

```

import network
import time
from machine import Pin
import time

led1 = Pin(2, Pin.OUT)
led2 = Pin(16, Pin.OUT)

ID = 1
NODES = {}
DEFAULT_PORT = '8080'

def print_and_blink():
    print('SinESP 1.0')
    print('Node {}'.format(ID))
    for i in range(ID):
        led1.on()
        time.sleep(0.5)
        led1.off()
        time.sleep(0.5)

def do_connect():
    sta_if = network.WLAN(network.STA_IF)
    ap_if = network.WLAN(network.AP_IF)
    ap_if.active(True)
    ap_if.config(essid='SinESP_WLAN1')
    ap_if.config(authmode=3, password='acdc1234')
    ap_if.ifconfig(('192.168.{}.1'.format(ID), '255.255.255.0',
'192.168.{}.1'.format(ID), '8.8.8.8'))

```

```

print('Habilitando rede Wi-Fi interna:', ap_if.ifconfig())
if not sta_if.isconnected():
    end_time = time.time() + 10
    countTimer = 0
    sleepTime = 0.500
    print('Conectando a rede de infraestrutura...')
    sta_if.active(True)
    sta_if.connect('BuildIII', 'vakonas3')
    while time.time() < end_time:
        time.sleep(sleepTime)
        countTimer += sleepTime
    if not sta_if.isconnected():
        print('Rede de infraestrutura não encontrada.')
        print('Conectando ao node mais próximo...')
        sta_if.active(True)
        sta_if.connect('SinESP_WLAN1', 'acdc1234')
        while not sta_if.isconnected():
            pass
    print('Conectado à rede:', sta_if.ifconfig())
    led1.on() #Por estar conectado ao GND e ao pin GPIO02, ao ligar o pino na
    verdade você o está desligando
    led2.off()

if __name__ == '__main__':
    print_and_blink()
    do_connect()

```

Arquivo main.py

```

import socket
import time
import sys
import network
import random
from boot import ID

HTML = """<!DOCTYPE html>
<html>
  <head> <title>{title}</title> </head>
  <body> <h1>{header}</h1>
    {content}
  </body>
</html>"""

DEFAULT_PORT = '8080'
NODES = {}
NODES_DATA = {}

#NODES = {1: '192.168.0.9:8000', 2: 'localhost:8001', 3: 'localhost:8002'}
#NODES = {1: 'localhost:8000', 2: 'localhost:8001', 3: 'localhost:8002'}

def update_node_data(node, value):
    global NODES_DATA
    print('updating locally node {}: {}'.format(node, value))
    node = int(node)
    value = float(value)
    NODES_DATA[node] = value
    x = 0.0
    c = 0

```

```

print('calculating mean...')
for node, i in NODES_DATA.items():
    if node == '__MEAN__':
        continue
    x += i
    c += 1
NODES_DATA['__MEAN__'] = x/(c or 1)
print('new node mean: {}'.format(NODES_DATA['__MEAN__']))

def http_client(_addr, _port, path):
    _port = int(_port)
    s = socket.socket()
    #ai = socket.getaddrinfo(_addr, _port)
    #addr = ai[0][-1]
    try:
        s.connect((_addr, _port))
    except:
        print('ERROR: cannot connect to {}:{}'.format(_addr, _port))
        s.close()
        return
    to_send = "GET {} HTTP/1.0\r\n\r\n".format(path)
    s.send(bytes(to_send, 'utf-8'))
    print('DEBUG: response from {}: {}'.format(_addr, s.recv(4096)))
    s.close()

def sensor(next_run=None):
    if next_run and next_run > time.time():
        return None
    data = random.uniform(0, 255)
    for node, _addr in NODES.items():
        print('sending {} to node {}'.format(data, node))
        if node == ID:

```

```

        update_node_data(node, data)
        continue
    addr, port = _addr.split(':')
    path = '/update/{}/{}'.format(ID, data)
    http_client(addr, port, path)
return time.time() + 10

def http_server():
    global NODES
    _addr, _port = NODES[ID].split(':')
    addr = socket.getaddrinfo(_addr, int(_port))[0][-1]
    s = socket.socket()
    s.settimeout(5)
    s.bind(addr)
    s.listen(5)
    print('[[[ listening on {} ]]]\n'.format(addr))

    next_run = None

    while True:
        has_next_run = sensor(next_run)
        if has_next_run:
            next_run = has_next_run
        try:
            cl, addr = s.accept()
            cl.settimeout(20)
        except Exception as e:
            print('ERROR: cannot receive connection from {}: {}'.format(addr, e))
            continue

        print('[connection w/ {} started]'.format(addr))

```

```

req_built = False
req = b''
cl_file = cl.makefile('rwb', 0)
while True:
    try:
        line = cl_file.readline()
    except Exception as e:
        print(e)
        break
    if not line or line == b'\r\n':
        req_built = True
        break
    req += line

if not req_built:
    print('invalid request: {}'.format(req))
    continue

method, path, _ = (i.decode() for i in req.split()[:3])

print('\nserving:\nmethod={} path={}\n\nresponse:'.format(method, path))

if method == 'GET' and path == '/data':
    c = '''<h2>DATA</h2>
        <div>{}</div>
        <h2>NODES</h2>
        <div>{}</div>'''.format(NODES_DATA, NODES)
    resp = HTML.format(title='', header='', content=c)

elif method == 'GET' and path.startswith('/update/'):
    data = path.replace('/update/', '')
    node, value = data.split('/')

```

```

        NODES[int(node)] = '{}:{}'.format(addr, DEFAULT_PORT)
        update_node_data(node, value)
        c = 'updated: node={} value={}'.format(node, value)
        resp = HTML.format(title='', header='', content=c)
    else:
        c = 'path "{}" not found'.format(path)
        resp = HTML.format(title='', header='', content=c)

    print(resp)
    #cl.send(bytes('HTTP/1.0 200 OK\r\n', 'utf-8'))
    cl.send(bytes(resp, 'utf-8'))
    cl.close()
    print('[connection w/ {} closed]'.format(addr))

def start():
    global NODES
    sta_if = network.WLAN(network.STA_IF)
    ap_if = network.WLAN(network.AP_IF)
    ifconfig_res = ap_if.ifconfig()
    ip, _, _, _ = ifconfig_res
    ifconfig_res = sta_if.ifconfig()
    _, _, gateway, _ = ifconfig_res
    gateway_node = gateway.split('.')[0]
    NODES[int(gateway_node)] = '{}:{}'.format(gateway, DEFAULT_PORT)
    NODES[ID] = '{}:{}'.format(ip, DEFAULT_PORT)
    http_server()

if __name__ == '__main__':
    start()

```