RELATORIO DE IMPLEMENTAÇÃO DO PROTOCOLO MQTT-SN

Randy Ramos Plácido*

RESUMO
Com o avanço na área de *IOT(Internet Of Things)* tem a necessidade de saber qual protocolo é mais estável e seguro para se utilizar nas trocas de pacotes, contendo informações, sem perder dados do pacote. Com esse objetivo, o tema do projeto é a comparação de protocolos de comunicação. Para ver como o protocolo reage com a interferência na rede.
O objetivo desse manual é guiar o desenvolvedor a implementar o protocolo MQTT-SN, e dar continuidade ao projeto de comparação entre os protocolos de comunicação MQTT, CoAP, MQTT-SN e AMQP.

1 **Materiais Utilizados**

Neste documento detalharemos os passos feitos no decorrer do semestre, para realizar o processo de implementação do protocolo MQTT-SN, precisaremos de alguns materiais para que ocorra a mesma. Logo abaixo segue uma lista de itens:

- 1 Módulo Esp8266 Nodemcu V3 *Wifi* 802.11;
- 1 Placa Raspberry Pi 3 Model B+;
- 1 cartão microSD de pelo menos 8Gb com a imagem do sistema previamente gravada;
- 1 cabo micro USB, para ligar o Modulo Esp8266 Nodemcu;

Após ter esses materiais, o próximo passo é fazer os Downloads das plataformas de desenvolvimento que foram utilizados para a implementação.

2 **Downloads e instalação**

Para o andamento do projeto há a necessidade de utilizar alguns programas de desenvolvimento.

2.1 Downloads

- IDE Arduino
- Biblioteca arduino-mqtt-sn-client
- Cliente Eclipse Paho MQTT-SN C / C ++ para plataformas embarcadas
- Mosquitto Broker
- Imagem do Raspberry Pi
- Etcher

Link para download da IDE Arduino:
https://www.arduino.cc/en/Main/Software
Link para download da biblioteca mqtt-sn-client:

* Departamento de Computação - Universidade Federal de Santa Catarina. Graduando em Engenharia de Computação. Universidade Federal de Santa Catarina. E-mail: Randy.placido@grad.ufs.br

https://github.com/S3ler/arduino-mqtt-sn-client

Link para download do programa Cliente Eclipse Paho MQTT-SN C / C ++ para plataformas embarcadas:

https://github.com/S3ler/arduino-mqtt-sn-client

Link para download da versão Raspbian Desktop:

https://www.raspberrypi.org/downloads/raspbian/

Link para download do Etcher:

https://www.balena.io/etcher/

2.2 **Instalações do Arduino e suas dependências**

Após baixar a IDE do Arduino e instalar é preciso fazer algumas alterações na mesma, para que possa trabalhar com a placa de desenvolvimento Esp8266.

Ao abrir a IDE do Arduino, primeiramente você irá ir em Arquivo > Preferências > Configurações > URLs adicionas para gerenciadores de placas > e adicionar a seguinte URL >http://arduino.esp8266.com/stable/package_esp8266com_index.json > OK. Conforme (figura 1).

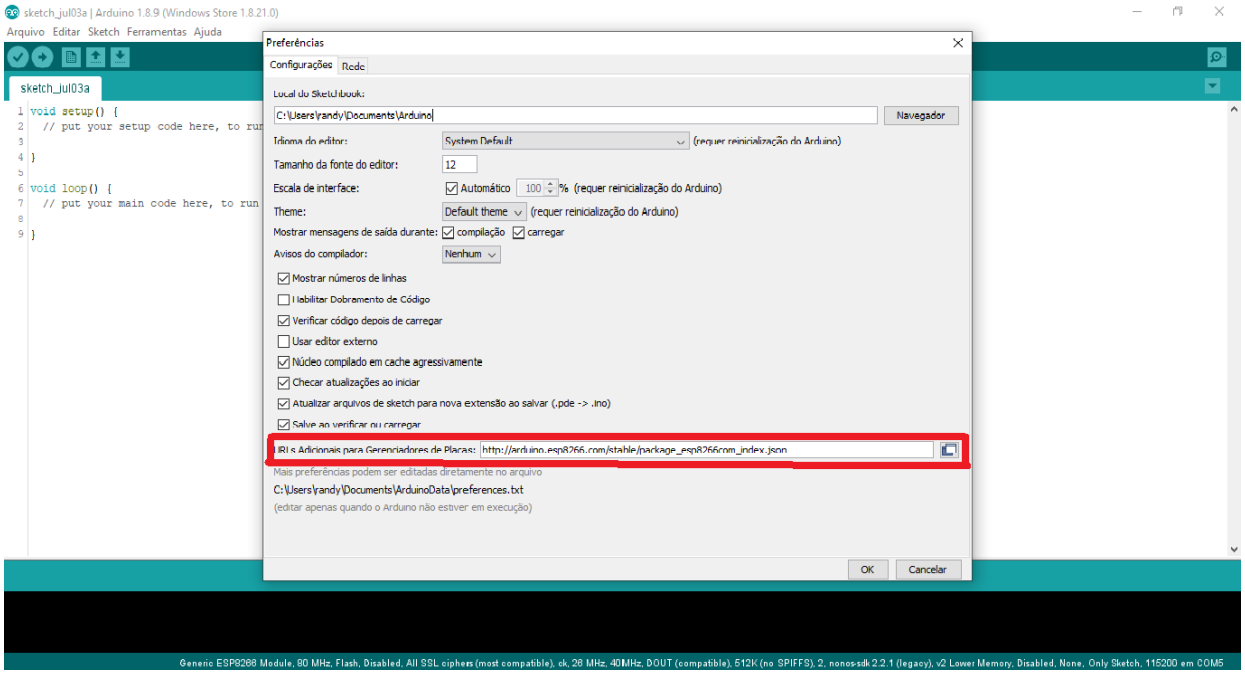

Figura 1 - Tela de configuração

O próximo passo será instalar a Biblioteca da placa m seguindo para a aba Ferramentas > Placa > Gerenciamento de placas > Pesquisar Esp8266 > Selecionar a versão > Instalar > Fechar. Conforme (Figura 2).

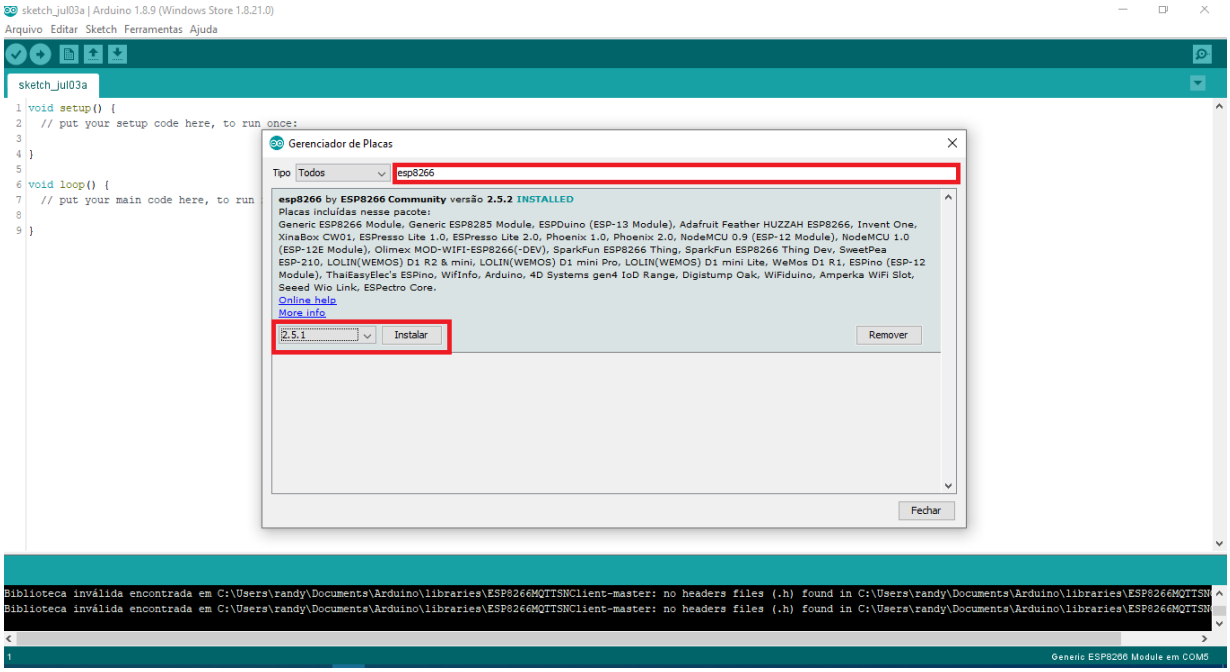

Figura 2 - Tela de gerenciamento de placas.

Em seguida, alterar nas configurações de placas, qual a placa que será utilizada, então terá que ir em Ferramentas > Placas > Selecionar *Generec* ESP82266 Module. Como visto (figura 3).

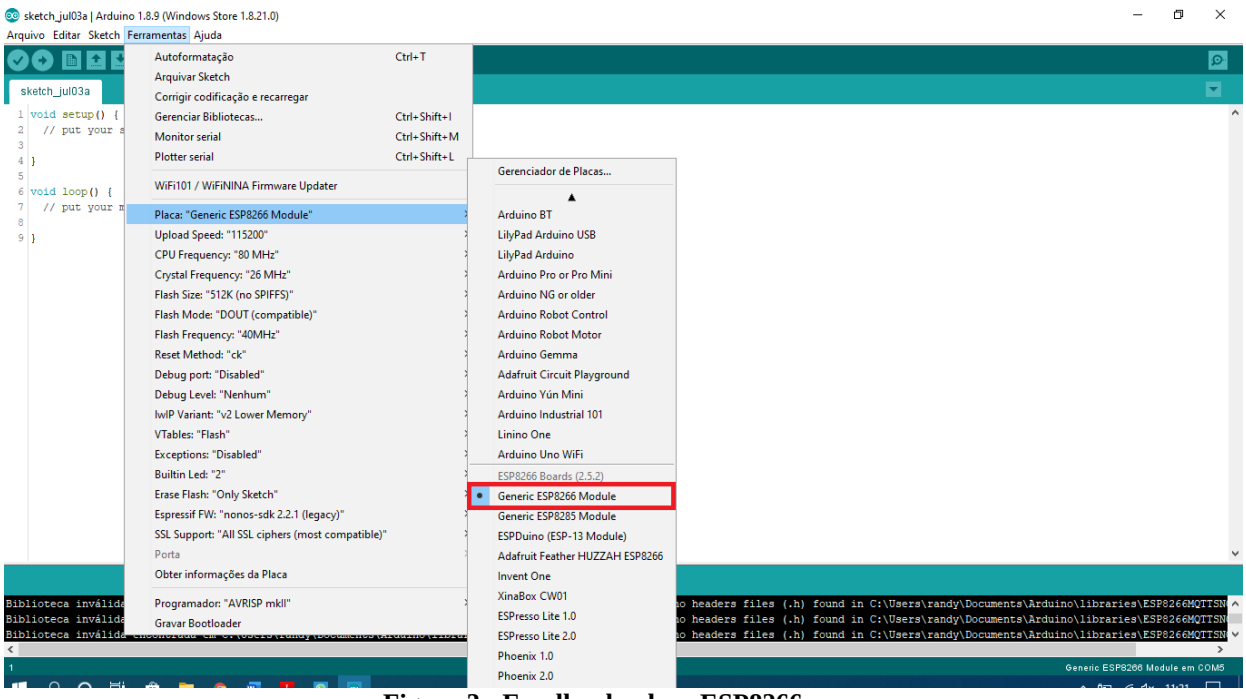

Figura 3 - Escolha da placa ESP8266

O Próximo passo será instalar a biblioteca do *MQTT-SN*, com a biblioteca baixada extraia o arquivo e renomeie a pasta para arduino-mqtt-sn-client e copia para o diretório da biblioteca do Arduino, reinicie seu Arduino IDE se estiver aberto.

Com isso, abra o exemplo em Arquivo > Exemplos > Arduin-mqtt-sn-client > Esp8266 > WiFiUdpMqttSnClient. Conforme (figura 4).

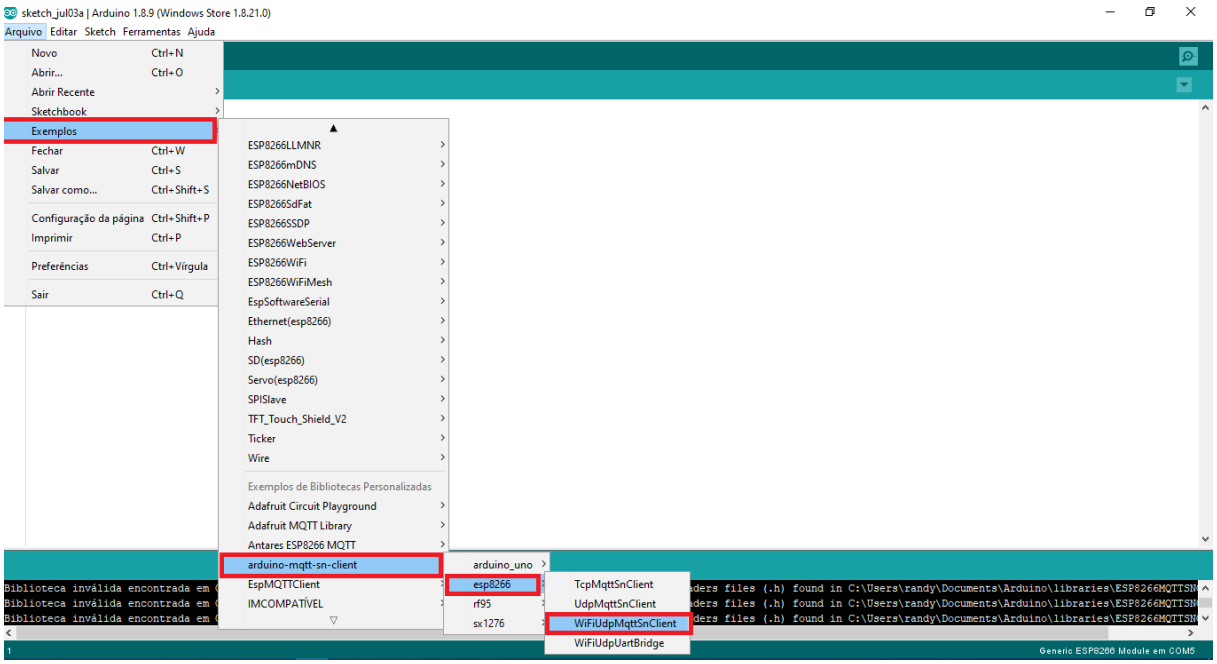

Figura 4 - Código do MQTT-SN

Adaptar *ssid* e *password* à sua rede *WiFi*, alterar o *gatewayIPAddress* e o *localUdpPort* para a porta *IPAddress* e *UDP* do seu *gateway MQTT-SN*. Como pode ser visto na (figura 5).

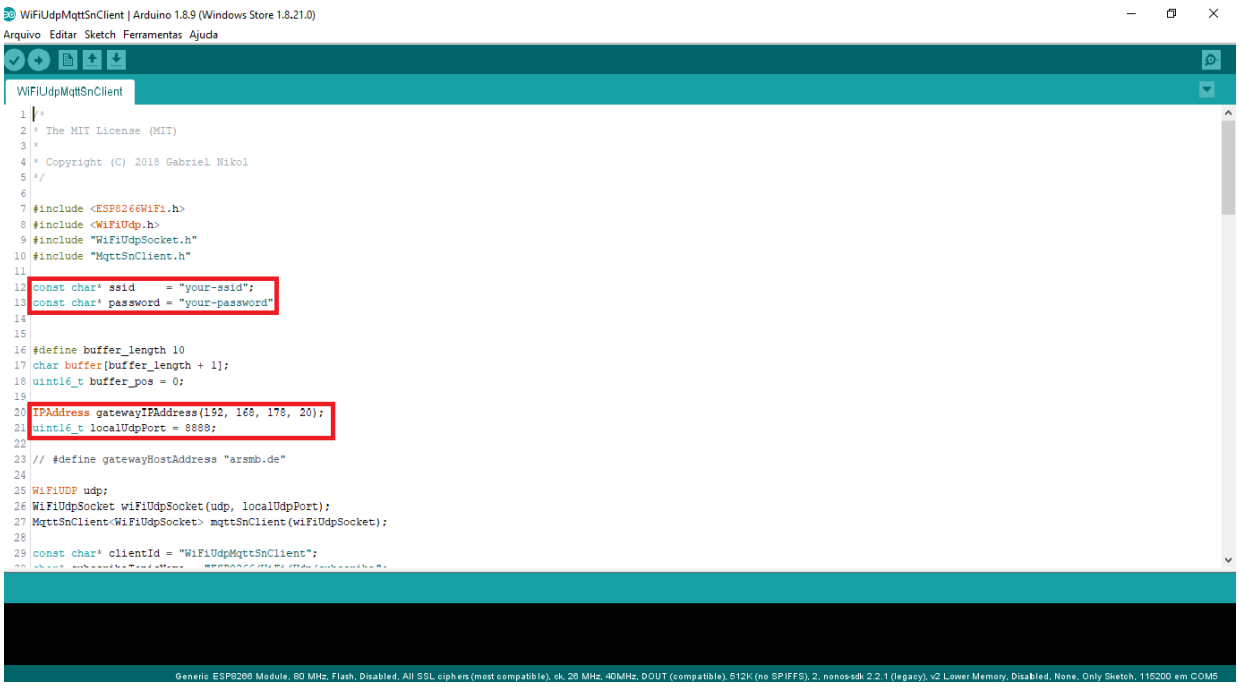

Figura 5 - Código WIFIUDMQTTSN-CLIENT

Por fim faça o upload do código para a placa e faça os testes e altere o código com os parâmetros que você definir.

O próximo passo que veremos a seguir será a configuração do *Raspberry Pi* para trabalhar como o *Gateway*, onde o *ESP8266* vai se comunicar com o *Gateway*, no qual o *Gateway* vai transformar o protocolo em *MQTT-SN* em *MQTT* para no final se comunicar com o *broker*.

2.3 **Instalação e Configuração do Sistema Operacional Raspbian**

Para instalar a *ISO* do *Raspbian* no cartão memoria, utilizaremos o programa Etcher. O Etcher é um programa para instalar de maneira simples o sistema operacional desejado em um carto SD.

Depois de baixar e instalar o *software* Etcher vamos selecionar a imagem clicando em *Select image* > selecionar seu cartão SD em *Seletc drive* > e por fim *click* em para iniciar o processo de gravação da imagem no cartão SD. Conforme (figura 6).

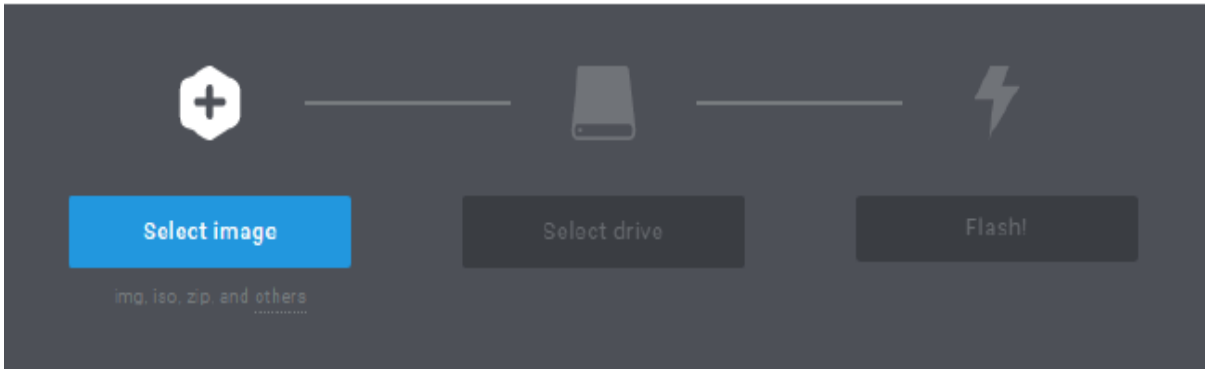

Figura 6 - Tela do Etcher

Com o sistema instalado no cartão SD, coloque-o no *raspberry* e conecte-o a um monitor, com o teclado e o mouse.

Feito isso o próximo passo será a configuração do sistema operacional. Na primeira inicialização, terá uma janela para configurações iniciais de idioma, fuso horário, senha de usuário, rede e atualizações.

- *Country: Brazil*
- *Language: Brazilian Portuguese*
- *Timezone*: São Paulo
- *Password: raspberry*
- Rede *Wifi*: conecte-se em sua rede ou pule clicando em *skip*.
- Update: clique em *next* e espere o sistema atualizar.
- *Reboot*: vai reiniciar o sistema e aplicar as devidas mudanças.

2.3.1 **Instalando o Cliente Eclipse Paho MQTT-SN C / C ++ para plataformas embarcadas**

Com o sistema devidamente instalado e atualizado, faremos construção do cliente *Paho MQTT-SN.* Para instalar o Cliente, necessitamos dar alguns comandos no terminal de comando do *Raspbian.*

O primeiro passo será entrar na pasta documentos para baixar o cliente Paho.

- *cd Documents*

Após estar dentro da pasta *Documents* daremos mais alguns comandos,

- *git clone https://github.com/eclipse/paho.mqtt-sn.embedded-c.git*

O próximo passo será construir o *Gateway.*

- *cd paho.mqtt-sn.embedded-c/MQTTSNGateway*
- *make*
- *make install*
- *make clean*

Com isso teremos que alterar as configurações do *Gateway* para o mesmo poder se conectar com o *Broker*.

- *cd Documents*
- *nano gateway.conf*

Com o *gateway.conf* aberto, altere o *BrokerName* para o *IP* do seu *Broker.* Conforme (figura 7)

```
# arquivo de configuração do MQTT-SN Gateway
#

BrokerName = iot.eclipse.org
BrokerPortNo = 1883
BrokerSecurePortNo = 8883

#
# Quando AggregatingGateway = YES ou ClientAuthentication = YES,
# Todos os clientes devem ser especificados pelo arquivo ClientList
#

ClientAuthentication = NÃO
AggregatingGateway = NO
QoS-1 = NÃO
Remetente = NÃO

# ClientsList = / path / to / your_clients.conf

PredefinidoTopic = NÃO
# PredefinedTopicList = / path / to / your_predefinedTopic.conf

# RootCAfile = / etc / ssl / certs / ca-certificates.crt
# RootCApath = / etc / ssl / certs /
# CertsFile = / path / to / certKey.pem
# PrivateKey = / path / to / privateKey.pem

GatewayID = 1
```

Figura 7 - Configuração do Client Paho Gateway

Com as devidas configurações feitas no arquivo, para salvar dar um Ctrl+ O > enter > Ctrl + X.

Então executar o programa com o seguinte comando:

- . /MQTT-SNGateway

Para finalizar a implementação falta instalar o broker mosquitto que veremos a seguir.

2.4 **Instalando o Broker Mosquitto nas dependências do Linux Debian**

O primeiro passo será atualizar os pacotes do Linux/Debian. Com os seguintes comandos no terminal.

- *sudo apt-get update*
- *sudo apt-get upgrade*

Em seguida, faça a instalação do software mosquitto

- *sudo apt-get install mosquitto*
- *sudo apt-get install mosquitto-clients*

Para finalizar configure o mosquito e inicie no com o seguinte comando mosquito -p 1883.

2.4 **Considerações finais e futuras implementações.**

No decorrer do projeto foram definidas algumas diretrizes a seguir, sendo primeiramente a estudar o protocolo a ser implementando, após o estudo partiria para a fase de implementação do protocolo de comunicação, após a implementação seria a etapa de teste, onde ficou definido que o primeiro teste seria o de vazão, e posteriormente o teste de bateria. Para esse momento ficou só o teste de vazão para ser feito e a última etapa seria comparar os testes para obtenção do resultado de qual protocolo se mantem estável.

A implementação do protocolo MQTT-SN ficou incompleto por parte do aluno, sendo preciso rever a parte do broker e dos testes de vazão, uma consideração para futuras implementações pelos acadêmicos, seria rever a parte do *Gateway* e cliente Eclipse Paho, estudar mais afundo os códigos de funcionamento do mesmo, ver o porquê de o Gateway não consegui fazer a publicação do cliente no tópico. Tentar usar o broker RSMB, e outra sugestão seria pesquisar se há a possibilidade de fazer a implementação com o Xbee Zigbee.