Synthèse documentations LoRa

ACHBAH Fatima-Zahra

OPTECH Casablanca, Maroc

Table des matières

Table des figures

1. Principe

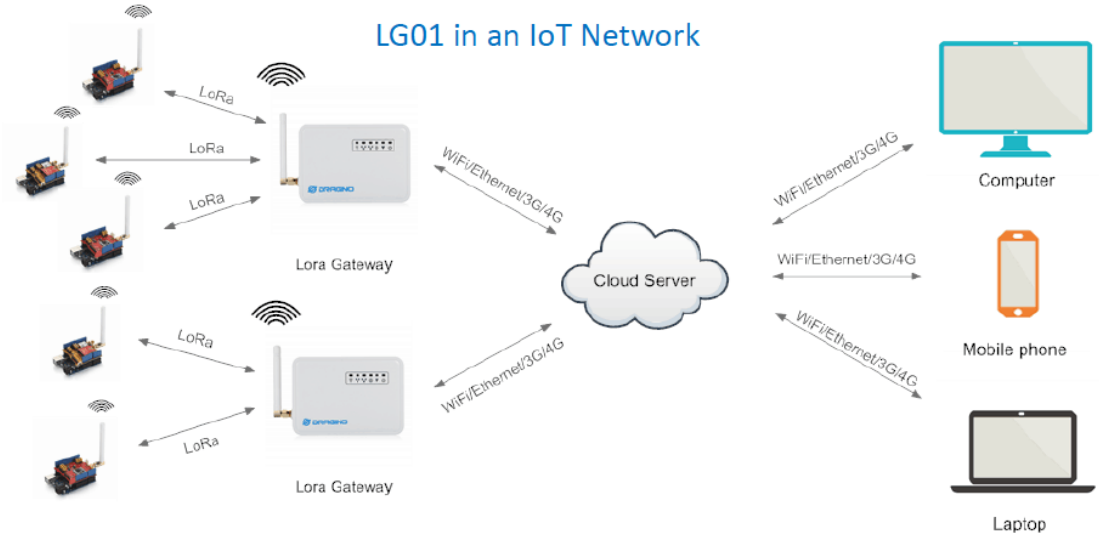

Figure 1: Schéma de Principe

2. Vue d'ensemble du système LG01-P

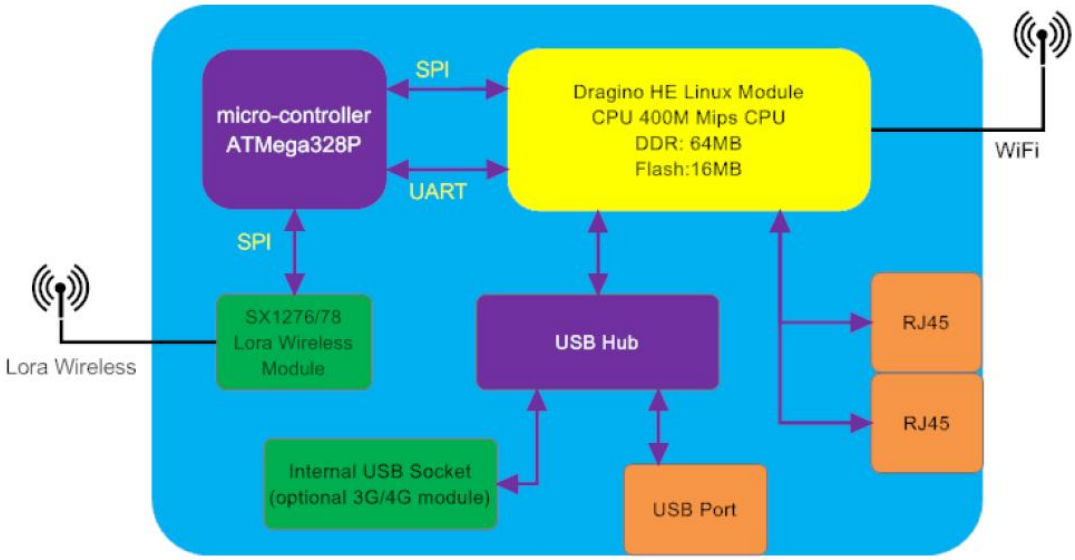

Figure 2: System overview

3. Fonctionnalités

Inclus un Linux Open source (OpenWrt). L'utilisateur peut modifier ou compiler le Firmware avec des spécifications personnalisées et sa propre marque.

- Basse consommation d'énergie.
- Compatible avec l'IDE 1.5.4 d'Arduino. L'utilisateur peut programmer, débugger ou téléverser le sketch du code sur le MCU via l'Arduino.
- Géré par Web GUI, SSHvia LAN ou WiFi
- Supporte la connexion internet via LAN, WiFi et 3G/4G.
- Serveur Internet intégré

- Auto-approvisionnement.

4. Spécifications

Système Hardware	Interface
Partie linux : ➢ Processeur ar9331 400Mhz ➢ RAM 64MB ➢ Flash 16MB Partie MCU : ➢ MCU : ATMega328P ➢ Flash : 32KB ➢ SRAM : 2KB ➢ EEPROM : 1KB	Power input : 9 ~ 24v DC 2 x RJ45 ports USB 2.0 Host port x 1 Internal USB 2.0 Host Interface x 1

Spécification Wifi	Spécification LoRa
IEEE 802.11 b/g/n Bande de fréquence: 2.4 ~ 2.462GHz Sensibilité du Wi-Fi : ➢ -11g 54M : -71dbm ➢ -11n 20M : -67dbm Energie Tx : ➢ 11n: mcs7/15: 11db mcs0 : 17db ➢ 11b: 18db ➢ 11g 54M: 12db ➢ 11g 6M :18db	Bande de fréquence: -Bande 1(HF) : 862-1020 Mhz -Bande 2(BF) : 410-528 Mhz 168 dB maximum link budget. ➢ +20 dBm - 100 mW constant RF output vs. ➢ +14 dBm high efficiency PA. ➢ Programmable bit rate up to 300 kbps. ➢ High sensitivity: down to -148 dBm. ➢ Bullet-proof front end: IIP3 = -12.5 dBm. ➢ Excellent blocking immunity. ➢ Low RX current of 10.3 mA, 200 nA register retention. ➢ Fully integrated synthesizer with a resolution of 61 Hz. ➢ FSK, GFSK, MSK, GMSK, LoRaTM and OOK modulation. ➢ Built-in bit synchronizer for clock recovery. ➢ Preamble detection. ➢ 127 dB Dynamic Range RSSI. ➢ Automatic RF Sense and CAD with ultra-fast AFC. ➢ Packet engine up to 256 bytes with CRC.

	➢ Built-in temperature sensor and low battery indicator.

Cellulaire 4G LTE	Cellulaire 3G UMTS/HSPA+
Quectel **EC20 LTE module** ➢ Micro SIM Slot ➢ Internal 4G Antenna + External 4G Sticker Antenna. ➢ Up to 100Mbps downlink and 50Mbps uplink data rates ➢ Worldwide LTE,UMTS/HSPA+ and GSM/GPRS/EDGE coverage ➢ MIMO technology meets demands for data rate and link reliability in modem wireless communication systems	Quectel **UC20 LTE module** ➢ Micro SIM Slot ➢ Internal 3G/4G Antenna + External 3G/4G Sticker Antenna. ➢ Up to 14.4Mbps downlink and 5.76Mbps uplink data rates ➢ Worldwide UMTS/HSPA+ and GSM/GPRS/EDGE coverage ➢ High-quality data and image transmission even in harsh environment ➢ Primary and diversity receive paths are designed for equivalent noise-figure performance

5. Applications

Figure 3: Dragino LoRa Gateway pour les applications IoT

6. Guide d'installation et manipulation rapide du Gateway

Veuillez Consulter la page : 12 du manuel
(http://www.dragino.com/downloads/downloads/UserManual/LG01_LoRa_Gateway_User_Manual.pdf)

7. Le setup du réseau

LG01 soutien des réseaux flexibles misent en place pour des environnements différents. Cette section décrit la topologie typique du réseau pouvant être définie dans LG01. Le réseau typique comprend:

- ✓ **Mode WAN Port Internet**
- ✓ **Mode Wi-Fi Client**
- ✓ **Mode Wi-Fi AP**
- ✓ **Mesh WiFi Network**
- ✓ **Mode USB Dial Up Mode**
- ✓ **Mode USB Ethernet**

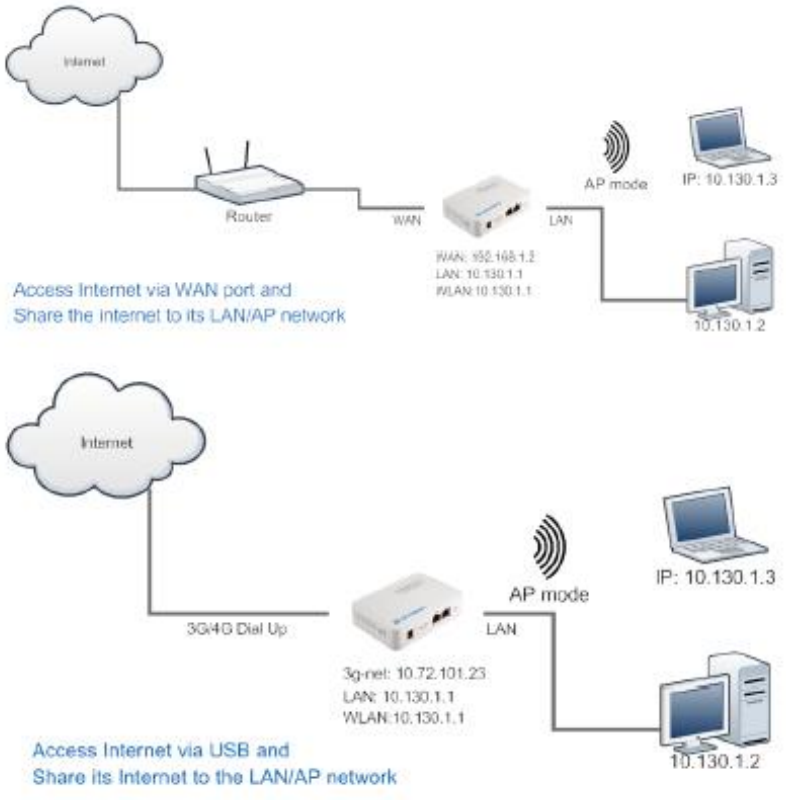

Figure 4: Réseau Wi-Fi AP

Pour plus de détails consulter à nouveau page 23 du manuel :
http://www.dragino.com/downloads/downloads/UserManual/LG01_LoRa_Gateway_User_Manual.pdf

8. Système Linux

LG01 se base sur le système Linux OpenWrt. Les utilisateurs peuvent configurer et ajuster leur système d'exploitation car il est en open source.

Pour accéder au console du Linux on utilise le protocole SSH. Pour les utilisateur de Windows ils peuvent utiliser les outils de SSH comme putty.

Pour ce faire vous auriez besoin de l'adresse IP de votre LG01. Le nom de l'utilisateur qui est dans ce cas : ***root*** et le mot de passe qui est : ***dragino***(par défaut).

LG01 supporte le **Protocol SCP**[1] et a un **serveur SFTP**[2] intégré. Il existe plusieurs façons d'éditer et de transférer des fichiers en utilisant ces deux protocoles. L'un des plus faciles est grâce à l'utilitaire ***WinSCP*** [1].

Après l'accès via WinSCP à l'appareil LG01, l'utilisateur peut utiliser une fenêtre similaire à FTP pour faire glisser / déposer ses fichiers sur le LG01 ou éditer les fichiers directement dans les fenêtres.

LG01 a une mémoire flash de 16 MB et une RAM de 64 MB. les deux fichiers /tmp /var changent continument à chaque redémarrage de l'appareil. Or le système Linux utilise 8MB-10 MB de la mémoire flash donc il ne reste pas beaucoup d'espace pour l'utilisateur pour stocker ses datas dans la mémoire flash de LG01. Mais l'utilisateur peut utiliser une mémoire externe comme l'USB.

9. La passerelle (Bridge)

La bibliothèque de la passerelle (Bridge) est la partie la plus importante de LG01. Car il fait la connexion entre la MCU (Partie Arduino)et le CPU (Partie ar9331).

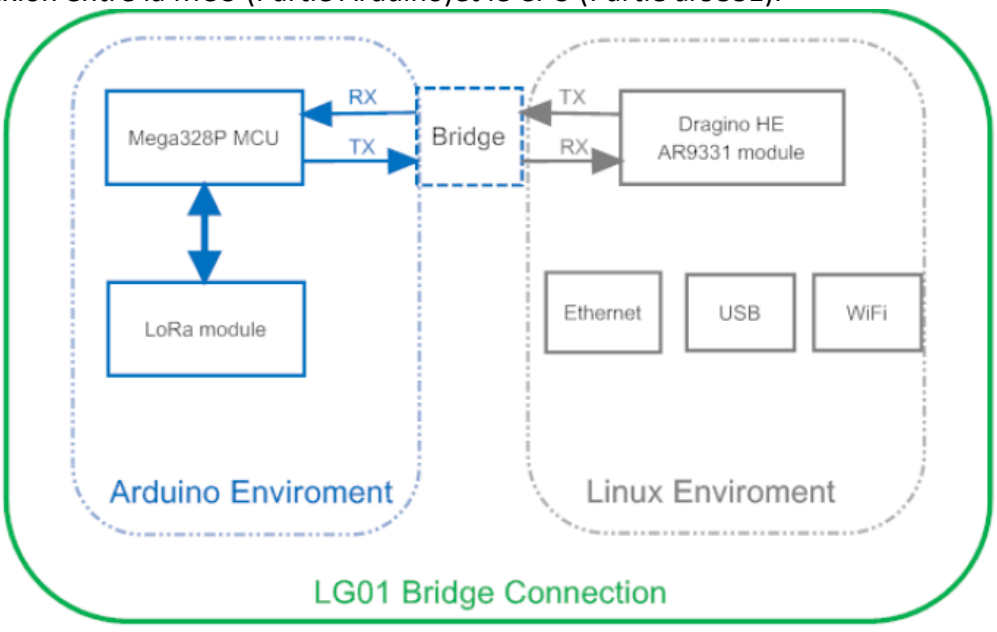

Figure 5: La passerelle entre le microcontrôleur ATMega328P et Linux

[1] ***Secure copy protocol*** *or* ***SCP*** *is a means of securely transferring computer files between a local host and a remote host or between two remote hosts. It is based on the Secure Shell (****SSH****) protocol. "SCP" commonly refers to both the Secure Copy Protocol and the program itself* [2]

[2] ***Secure File Transfer Protocol or SFTP****, which stands for SSH File Transfer Protocol, or Secure File Transfer Protocol, is a separate protocol packaged with SSH that works in a similar way over a secure connection. The advantage is the ability to leverage a secure connection to transfer files and traverse the filesystem on both the local and remote system. In almost all cases, SFTP is preferable to FTP because of its underlying security features and ability to piggy-back on an SSH connection.* [3]

La bibliothèque de la passerelle définit le mécanisme de communication entre le MCU et le CPU(ar9331). Avec cette bibliothèque l'utilisateur peut envoyer la data vers le CPU, obtenir des commandes résultat du CPU ou faire appel à des commandes dans le CPU. La bibliothèque du Bridge utilise le port UART pour communiquer entre MCU (Arduino) et ar9331.

10. Exemples d'applications

10.1. Intégrer LoRa avec l'API RESTful[3]

En réalité, ils existent différents types d'APIs mais dans le manuel il utilisent un API RESTful car le serveur ThingSpeak le supporte. Pour communiquer data de la borne on aura besoin d'un compte dans un serveur, créer une canal. l'ID de cette canal reste l'unique ID pour stocker nos datas dans le serveur. Après il faut faire communiquer le LG01 avec cette canal. On a aussi besoin des clés API et méthode API. (Pour plus d'infos consulter le manuel).

10.1.1. Teste de la liaison montante (Uplink)

Pour faire un Uplink on doit programmer LG01 pour téléverser la data du capteur au serveur. Dans le manuel il utilise le serveur ThingSpeak.

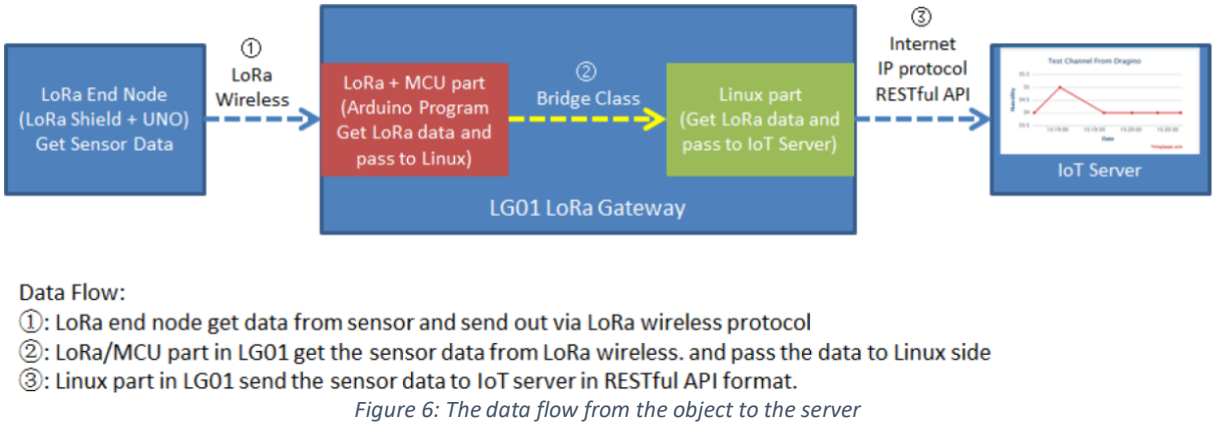

Figure 6: The data flow from the object to the server

Pour 1: Envoyer les datas du capteur au MCU avec le Protocol LoRa on doit programmer la partie Arduino du Gateway et le microcontrôleur au niveau du nœud. (page : 13-17)
Après il faut programmer le Bridge(la passerelle). Pour ce faire il faut aller voir le manuel page : 36.
Dans la suite de cette partie il faut intégrer les 3 étapes pour réussir à faire l'Uplink de Datas (page: 42).

- **Try Restful API call in web**

Il faut tout d'abord préciser le type d'API qu'on utilisera dans le serveur d'Optech. Après il faut faire appel à l'API pour mettre à jour le flux de canal(the channel feed). Dans l'appel de l'API on a 3 variables :
api_key: Définir sur quel canal vous allez téléverser les données
Field: Chaque canal a 8 Fields (champs) on doit préciser quel field à mettre à jour
Value: Field1=0 veut dire field1 est mis à jour avec la valeur 0
Refer to page: 45

[3] https://stackoverflow.com/questions/671118/what-exactly-is-restful-programming

- **Try RESTful API call with LG01 Linux command**

Il faut s'assurer que LG01 est connecte à un réseau internet. On peut se connecter au SSH et épingler une adresse internet et voir si ça marche ou pas. (Page 46).
LG01 a un outil Linux Curl intégré. Cet outil est très puissant pour une communication http. On peut utiliser cet outil pour gérer l'appel de RESTful API dans LG01.

la commande pour mettre à jour le feed :

curl -k "https://api.thingspeak.com/update?api_key=B9Z0R25QNVEBKIFY&field1=40" ("""" must be included) page(46)

The curl command is executed in the Linux side. Then we will have to call curl command with sensor data variable in Arduino side. This is through the process class in Arduino: https://www.arduino.cc/en/Tutorial/Process

(Le reste de cette documentation est en anglais)

- **Integrate LoRa, Bridge and Curl**

- LoRa End Node: LoRa Shield+UNO+DHT11. The LoRa End node keeps getting temperature and humidity from the sensor and sends out via LoRa periodically.

- LoRa Gateway LG01: Listening on the LoRa wireless channel, while there is new LoRa packet arrives, parse it and send out to IoT Server.

10.1.2. Downlink Test

We will try to program LG01 to fetch download data from ThingSpeak, then broadcast this data to local LoRa network. The end node will gets this message and check if they need to do something.

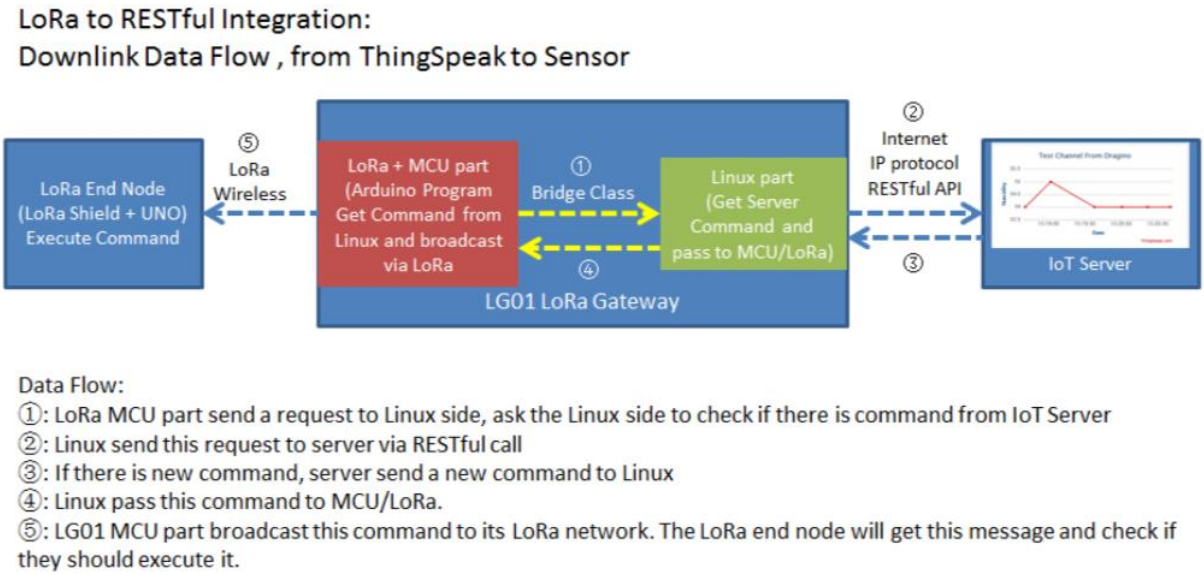

Figure 7: The data flow from server to the object (sensor)

We will try first to do it on PC, and then do it in Linux, and finally integrate it with LoRa.

- **Create Talkback command and try RESTful API call in web**

To do downlink test we need to first create a talkback command in ThingSpeak. As below, From this page, we can get the talkback API key and set the command to be sent to the LoRa End Device.

(page: 49)

- **Try RESTful API call with LG01 Linux command**

The command to be used is Execute the Next TalkBack Command, with curl, it is
curl -k "https://api.thingspeak.com/update?api_key=B9Z0R25QNVEBKIFY&field1=40"

Integrate LoRa, Bridge and Curl

10.2. Advance Examples

➢ Example for connecting to TTN LoRaWAN server

http://wiki.dragino.com/index.php?title=Connect_to_TTN

11. Advance Examples

11.1 Example for Connecting to TTN LoRaWAN server

Please check this link for detail: **Connect to TTN**

11.2 Multiple Nodes examples

The example shows how the gateway can handle multiple nodes up to several hundreds. The example can be found at **IDE --> File --> Examples --> Dragino --> LoRa --> Concurrent**

How it works:

This concurrent client sketch is working together with the concurrent gateway sketch. Before using this sketch, please use the write_client_id sketch to write a client ID in the EEPROM. Client ID is the unique id for each client in the LoRa network. write_gateway_id to gateway is not necessary, if not write, gateway id will be 0XFF.

➢ When the client boot, it will keep listening for the broadcast message from LoRa Gateway.

- When the gateway sketch boot, it will broadcast a message to set up a LoRa Network. If it gets broadcast message, client will send a join request message to gateway, when the join request message arrive to gateway, the gateway will send back a join-ack message with client id and add this client to the LoRa Network.
- If the client gets its join-ack message for its join request, it will enter the mode to listen the data-request message from gateway. In this mode, if client get a data-request message for this client it will send back a data message to the gateway.
- After client in data_request listening mode, if it has not receive any message from gateway in a timeout, it will go back to the network set up mode to listen the broadcast message.
- Gateway will refresh the LoRa network periodically for adding new client or remove unreachable client.

This example using the polling method between LoRa node and Gateway, it will minimize the LoRa packets transfer on the air and avoid congestion. It is suitable for a not real time LoRa work.

Performance test in a room with 100 nodes and 1 gateway shows:

a) Gateway require about 1.5 minutes to set up this 100 nodes Network

b) Gateway takes about 2 minutes to do polling for these 100 nodes.

User can adjust the timing in the sketch from case by case.

What kind of server the LG01 can support ?

The Linux side of LG01 is **OpenWrt**, it is open source and users can develop application over it. Basically it can support most IoT servers if use the right API. We have examples for how to connect some servers via typical protocol (MQTT, RESTful) for IoT, MQTT or RESTful. From this link: https://github.com/dragino/Arduino-Profile-Examples/tree/master/libraries/Dragino/examples/IoTServer (IoT Server Examples).

13.2 Bridge between MCU and Linux module doesn't work.

Some possibilities:

1/ You have used the Serial class in MCU sketch, like Serial.begin(9600), The bridge library in Mega328P use the same Serial interface. So if you have the Serial code in the sketch. They will conflict and bridge doesn't work.

2/ The IDE get mess in the serial setting when you compile other sketch . In this case, you can close the IDE and open it again.

Bibliographie

[1] [En ligne]. Available: https://en.wikipedia.org/wiki/WinSCP.

[2] [En ligne]. Available: https://en.wikipedia.org/wiki/Secure_copy.

[3] [En ligne]. Available: https://www.digitalocean.com/community/tutorials/how-to-use-sftp-to-securely-transfer-files-with-a-remote-server.