CAPITOLO 9

PARAGRAFO 6 Progetto finale[1]: la comunicazione dal sensore alla piattaforma IoT cloud

In questo paragrafo si analizza un esempio di impiego di un sistema di comunicazione per l'ambito IoT che consente di trasferire i dati generati da dei sensori montati su un *end node*, posizionabile sia in ambiente interno (*indoor*) sia in ambiente esterno (*outdoor*), a una piattaforma cloud, impiegando piattaforme di prototipazione, connessioni wireless LPWAN (*Low Power Wide Area Network*) e connessioni Internet.
L'esempio proposto utilizza la tecnologia wireless **LoRa**, i protocolli e un'infrastruttura di rete di tipo **LoRaWAN** e una normale connessione Internet.
LoRA (*Long Range*) è una tecnologia trasmissiva che consente connessioni wireless a lunga distanza (*Wide Area*), dell'ordine della decina di km in ambiente aperto, con dispositivi a basso consumo (*Low Power*) e a basso costo. Essa è classificabile come tecnologia LPWAN (*Low Power Wide Area Network*) impiegabile in applicazioni IoT dove è necessario avere molti sensori distribuiti sul territorio, i cui dati tipicamente vengono inviati ad applicazioni residenti su piattaforme cloud poste su Internet.
Per applicazioni di questo tipo a *LoRa* sono stati affiancati una topologia di rete a stella e appositi protocolli che consentono di inviare dati a server (detti *network server*) posti su Internet, i quali possono poi inoltrarli ad apposite piattaforme cloud IoT ove risiedono le applicazioni deputate alla raccolta dei dati e alla gestione dei sensori e dei relativi apparati.

L'insieme della tecnologia trasmissiva LoRA e dei protocolli appositamente sviluppati viene denominato **LoRaWAN**.

La struttura di riferimento di una rete LoRaWAN è costituita da una topologia a stella caratterizzata dai seguenti elementi:

- **end node**, tipicamente comprende dei sensori (e/o attuatori), un sistema a microcontrollore (come Arduino e suoi derivati), un modulo ricetrasmissivo wireless in tecnologia *LoRa*;
- **gateway (**o **concentratore**) connesso a Internet; raccoglie i dati inviati dagli *end node* via *LoRa* e li inoltra a un *network server* tramite una connessione Internet;
- **Network Server (**accessibile via Internet): raccoglie i dati inviati dagli *end node* tramite il gateway, gestisce gateway ed end node, può visualizzare i dati ed eventualmente inoltrarli ad altre piattaforme cloud IoT dove risiedono le applicazioni IoT impiegate per erogare i servizi resi disponibili dai dati raccolti.

[1] Si ringrazia il prof. Venanzio Cesari per il contributo dato nella fase di realizzazione del progetto proposto.

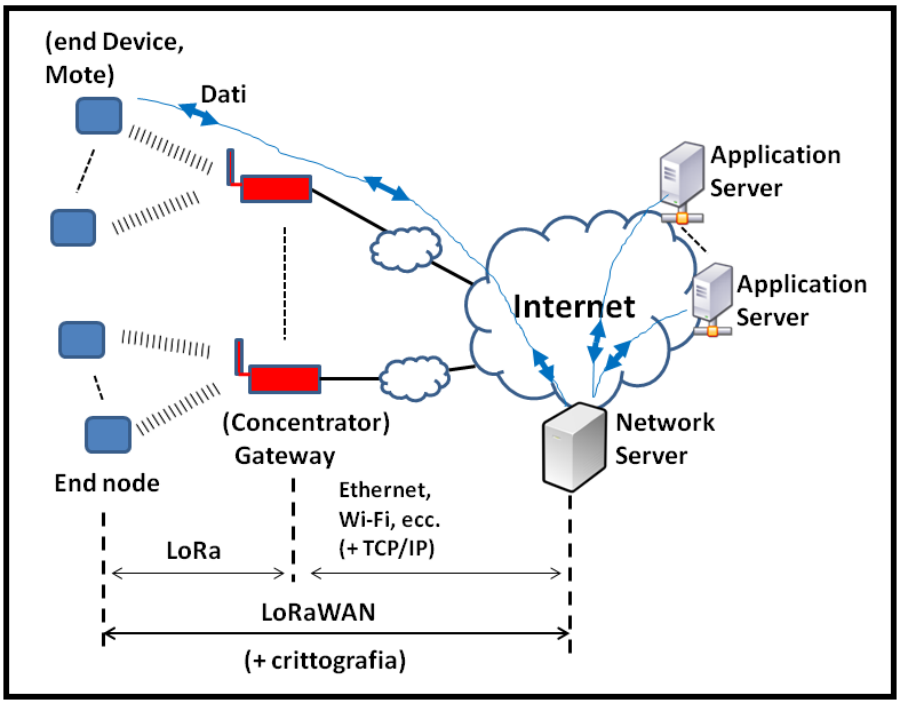


FIGURA 1 Schema di riferimento per l'impiego in ambito IoT di LoRaWAN

A titolo esemplificativo si illustra la realizzazione di una piccola rete LPWAN in tecnologia LoRaWAN, a basso costo, che ha visto l'impiego dei seguenti componenti:

- kit di sviluppo per IoT **Seeed LoRaWAN Gateway 868 MHz Kit**

(https://www.seeedstudio.com/LoRa-LoRaWAN-Gateway-868MHz-Kit-with-Raspberry-Pi-3-p-2823.html) che comprende i seguenti elementi:
  - **un end node** (detto anche *mote* o *end device*), costituito da un apparato Arduino-compatibile denominato **Seeeduino LoRaWAN**, che integra un modulo radio *LoRa*, è programmabile con l'IDE di Arduino e per il cui utilizzo viene fornita la libreria *LoRaWan.h*, scaricabile da Internet tramite l'IDE di Arduino; per realizzare e rendere operativo un end node è sufficiente collegare a Seeeduino dei sensori e programmarlo con sketch analoghi a quelli con cui si programma Arduino, in modo da acquisire i dati dai sensori e inviarli al gateway tramite la tecnologia wireless LoRa;
  - **un gateway** costituito da un microcomputer **Raspberry Pi 3**, dotato di SD su cui è già installato il software[2] per la comunicazione via Internet con un *Network Server LoRaWAN* della piattaforma cloud per IoT **LORIOT.io** e da un modulo radio per la comunicazione via *LoRa* con gli *end node*;
- un secondo **end node**, di tipo diverso, costituito da un dispositivo **Arduino MKR WAN 1300**, che integra già un modulo radio *LoRa* e si programma con l'IDE di Arduino;
- la **piattaforma cloud LORIOT.io** (www.loriot.io), che offre sia il servizio di *Network Server* LoRaWAN sia la possibilità di inoltrare i dati raccolti ad altre piattaforme/servizi cloud, come *AllthingsTalk.com* (www.allthingstalk.com/maker) con svariate modalità; la registrazione a *LORIOT.io* è gratuita per la realizzazione di una rete LoRaWAN costituita da un gateway e da un massimo di 10 end node.

[2] Nel caso si desideri effettuare una propria installazione del software, una volta registrati alla piattaforma Loriot, come indicato nel paragrafo 6.5, selezionando nella dashboard il proprio Gateway è possibile cliccare su Software per ottenere le informazioni necessarie all'installazione del software che rende Raspberry Pi un Gateway Loriot

Per semplicità e a scopo dimostrativo negli end node si impiegano un sensore di temperatura LM35 e un sensore di temperatura e umidità DHT11 direttamente collegati. Il sistema che si viene a realizzare è rappresentato in FIGURA 2.

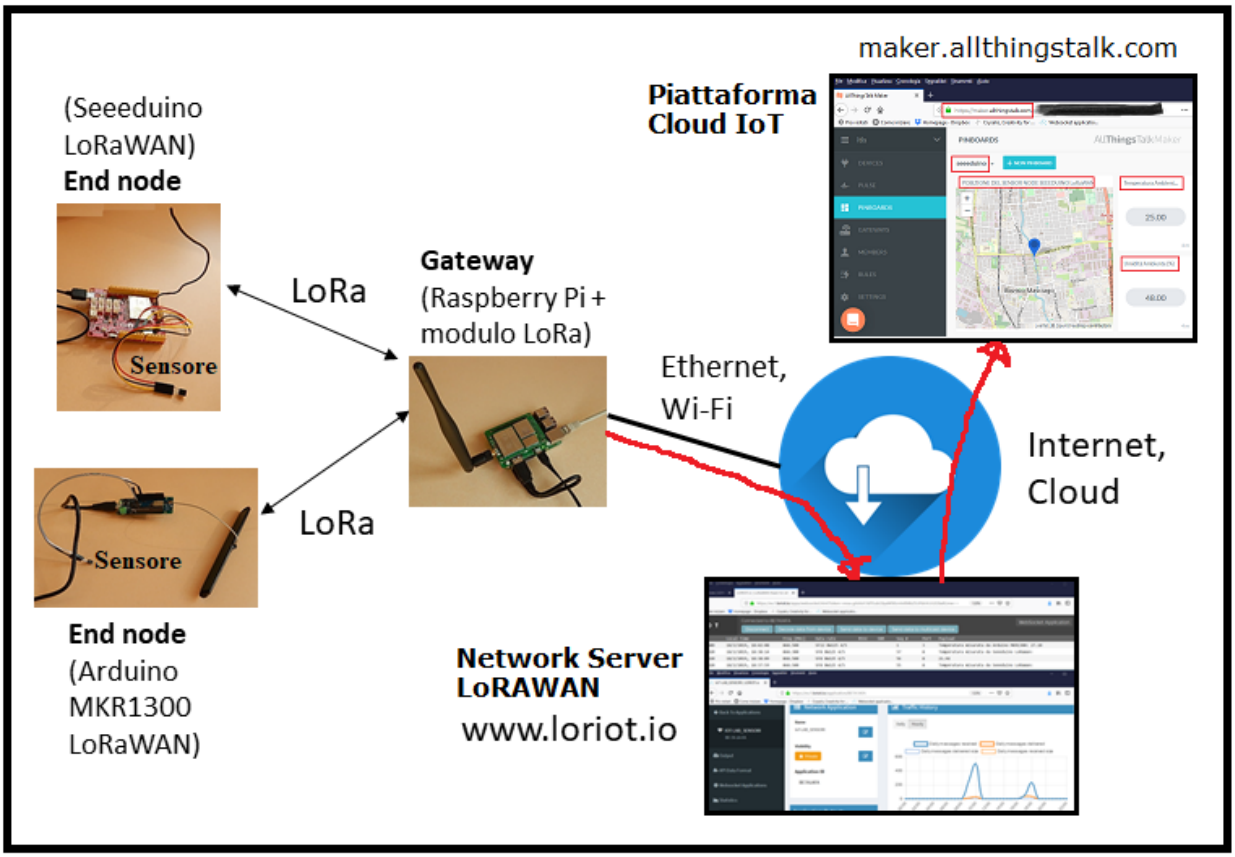


FIGURA 2 Componenti impiegati per esemplificare un'applicazione IoT che usa la tecnologia LoRaWAN.

Dal punto di vista dell'impiego pratico, le caratteristiche fondamentali del sistema LoRaWAN sono qui di seguito riportate.
In Europa la tecnologia LoRa può operare nelle bande di frequenza libere (ISM) attorno a 868 MHz e a 433 MHz; qui si illustrano solo le caratteristiche del sistema impiegato, che opera a 868 MHz, e che sono le seguenti:

- la potenza di trasmissione, espressa in termini di EIRP, è pari a $\mathbf{EIRP_{max} = 14\ dBm}$;
- l'organizzazione dei canali viene denominata **EU868** ed è riportata in TABELLA 2;
- Il *data rate* (bit rate) può essere calcolato con la seguente relazione:

$$DR = SF \cdot \frac{BW}{2^{SF}} \cdot CR \qquad (1)$$

dove SF è il fattore di spreading, BW è la banda di canale, CR è il Code Rate impiegato per la correzione d'errore di tipo FEC (*Forward Error Correction*).
Poiché il valore del fattore di spreading SF è compreso tra 7 e 12, con i valori tipici di banda (BW) e Code Rate (CR), pari a BW=125 kHz e CR=4/5, i valori dei principali data rate risultano[3] quelli riportati in Tabella 1, identificati con un numero compreso tra 0 e 5;

[3] Con SF=7 per il canale a 806.3 MHz è ammessa anche una banda di 250 kHz che consente di ottenere un data rate di circa 11 kbit/s. Inoltre a 868.8 MHz è previsto anche l'impiego della modulazione di frequenza FSK che con banda di canale 250 consente data rate pari a circa 50 kbit/s.

- a livello applicazione, la dimensione massima del *payload* (campo dati) dipende dal *data rate* con cui si opera e varia da *51 Byte*, con DR0 (circa 300 bit/s) a *222 Byte* con DR5 (circa 5500 bit/s).

Tabella 1 Data Rate (DR) supportati da LoRaWAN per canali con banda 125 kHz

SF (Spreading Factor)	BW [Hz] (Banda)	CR (Code Rate= 4/5)	Valore del DR [bit/s]	Identificativo del Data Rate (DR)
12	125000	0,8	293	0
11	125000	0,8	537	1
10	125000	0,8	977	2
9	125000	0,8	1758	3
8	125000	0,8	3125	4
7	125000	0,8	5469	5

Per quanto concerne i canali radio impiegati dal kit utilizzato, denominati CH0-CH7, essi sono conformi alla pianificazione EU868 (standard LoRAWAN 1.0x) e in trasmissione[4] sono i seguenti:

Tabella 2 Principali canali radio EU868

Denominazione Canale	Frequenza [MHz]	Data Rate supportati in uplink
CH0	867.1	DR0÷DR5
CH1	867.3	DR0÷DR5
CH2	867.5	DR0÷DR5
CH3	867.7	DR0÷DR5
CH4	867.9	DR0÷DR5
CH5	868.1	DR0÷DR5
CH6	868.3	DR0÷DR5
CH7	868.5	DR0÷DR5

A titolo esemplificativo, in Figura 3 è mostrato lo spettro del segnale irradiato dall'end node Seeeduino LoRaWAN quando trasmette alla frequenza di 868,1 MHz.

[4] Lo standard prevede che i canali CH 5, 6, 7 debbano essere obbligatoriamente supportati dai dispositivi LoRa. In ricezione si ha un canale (RX2) a 869.525 MHz con banda 125 kHz e SF = 12

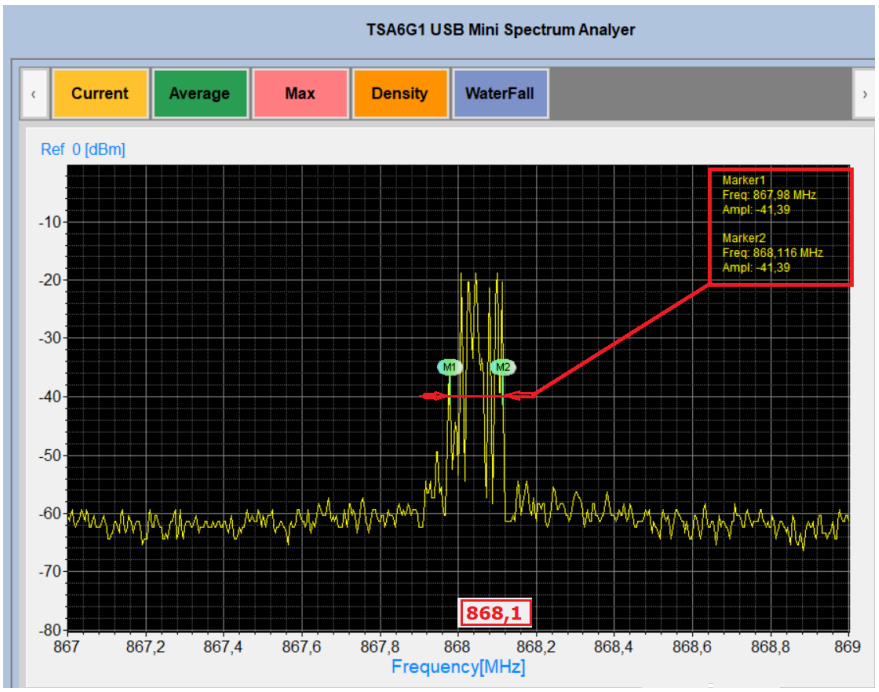


FIGURA 3 Spettro del segnale irradiato da Seeeduino LoRaWAN operante sul canale a 8068.1 MHz, con banda di canale 125 kHz.

6.1 Installazione del Gateway LoRaWAN

Il gateway LoRaWAN utilizzato (FIGURA 4) è costituito da

- un microcomputer Raspberry Pi 3, con sistema operativo, programmi applicativi e protocolli per comunicare con un *Network Server* LoRaWAN di LORIOT e con gli *end node*, ma che comprende anche un network server interno utilizzabile per dei test in locale;
- un modulo radio *LoRa* per la comunicazione con gli *end node*.

Seguendo le istruzioni fornite dal costruttore del kit si monta il modulo radio su Raspberry Pi. Si collega quindi in rete, via Ethernet, Raspberry Pi e se ne determina l'indirizzo IPv4, per esempio con uno dei seguenti metodi:

- collegandosi localmente con video, tastiera e mouse, e digitando il comando *ifconfig* (o *ip address*)
- impiegando un PC collegato in rete e lanciando un programma di scansione della rete come Zenmap.

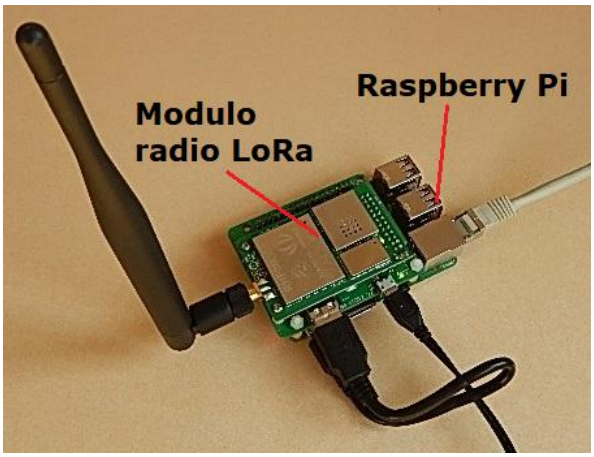


FIGURA 4 Gateway basato su Raspberry Pi

Noto l'indirizzo IP di Raspberry Pi è possibile collegarsi da remoto utilizzando un client SSH come PuTTY e impiegando le credenziali di accesso (username e password) indicate nella documentazione.

Di default è attivo il network server interno al gateway Raspberry Pi, accessibile via browser inserendo nella sua barra l'indirizzo IPv4 di Raspberry Pi (per verificare se il network server è effettivamente attivo seguire le indicazioni fornite dalla documentazione, inviando il comando `sudo systemctl status pktfwd`)
Dopo essersi registrati sulla piattaforma cloud LORIOT (www.loriot.io) e aver registrato il gateway, come indicato nel PARAGRAFO 6.5, per far collegare Raspberry Pi con il network server in cloud di LORIOT https://eu1.loriot.io è sufficiente (FIGURA 5):

- aprire un terminale su Raspberry Pi, per esempio da remoto via SSH con PuTTY

- fermare il server interno con il comando

```
sudo systemctl stop pktfwd
```

- resettare il gateway con il comando

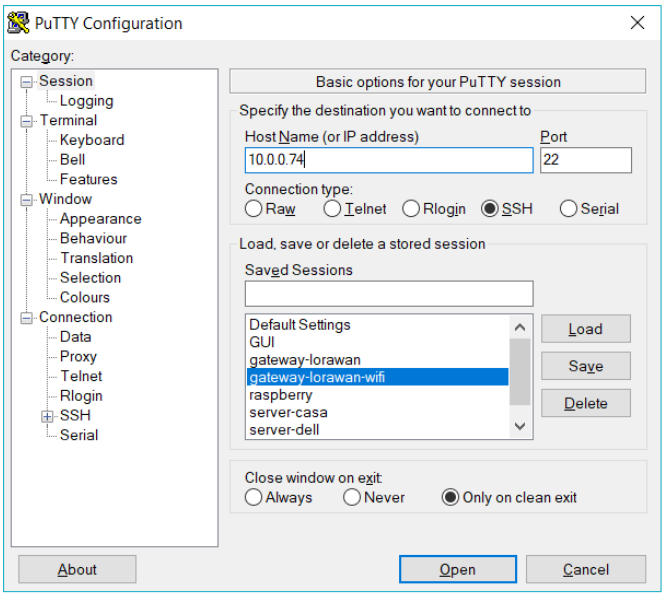
```
sudo gwrst
```

- entrare nella directory contente l'ultima versione del software che implementa lo standard LoRaWAN (al momento della stesura dell'esercitazione è la 1.0.2)

```
cd loriot/1.0.2
```

- digitare il comando (prestare attenzione all'indirizzo del network server inserito nel comando, che deve essere quello della registrazione sulla piattaforma LORIOT; nel nostro caso è `eu1.loriot.io`)

```
./loriot-gw.bin -f -s eu1.loriot.io
```



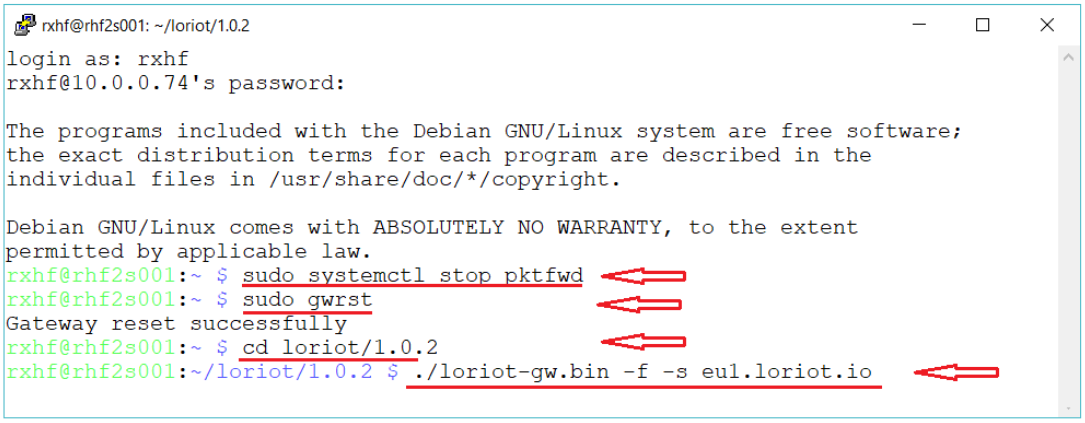


FIGURA 5 Comandi per far connettere il gateway con il network server LORIOT

Personalizzazione del gateway
Con la versione del software installato su Raspberry Pi al momento della stesura dell'esercitazione è possibile personalizzare Raspberry Pi in modo che:

1. all'accensione Raspberry Pi si colleghi automaticamente al Network Server in cloud di LORIOT (fermando prima il network server interno); a questo scopo è sufficiente (FIGURA 6) andare nella directory `etc` (comando: `cd /etc`) e aggiungere con l'edito di testo nano (comando `sudo nano rc.local`) al file **rc.local** (prima di *`exit 0`*) i comandi:

```
sudo systemctl stop pktfwd
sudo gwrst
/home/rxhf/loriot/1.0.2/loriot-gw.bin -f -s eu1.loriot.io
```

Salvare premendo i tasti `ctrl O` (control O), uscire premendo `ctrl X` e riavviare con il comando `sudo shutdown –r now` (oppure con `sudo reboot`).

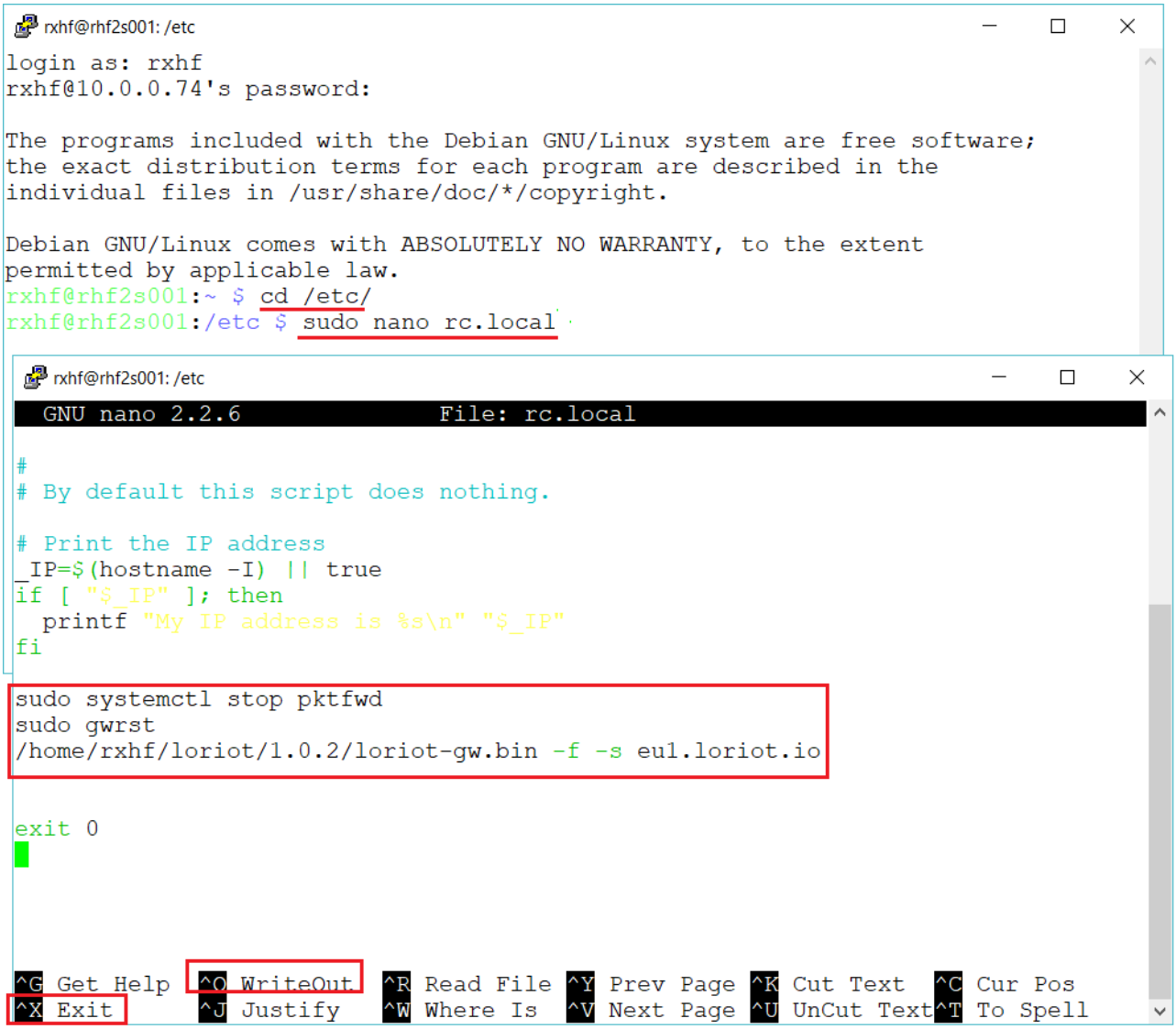


FIGURA 6 Configurazione dell'avvio automatico della connessione con Loriot.io

2. Raspberry Pi si colleghi in rete via Wi-Fi e non via Ethernet; a questo scopo è possibile[5] entrare nella directory `/etc/network` con il comando `cd /etc/network` , editare il file interfaces (`sudo nano interfaces`) come mostrato in FIGURA 7.
Salvare (ctrl O), uscire (ctrl X) e riavviare Raspberry Pi.

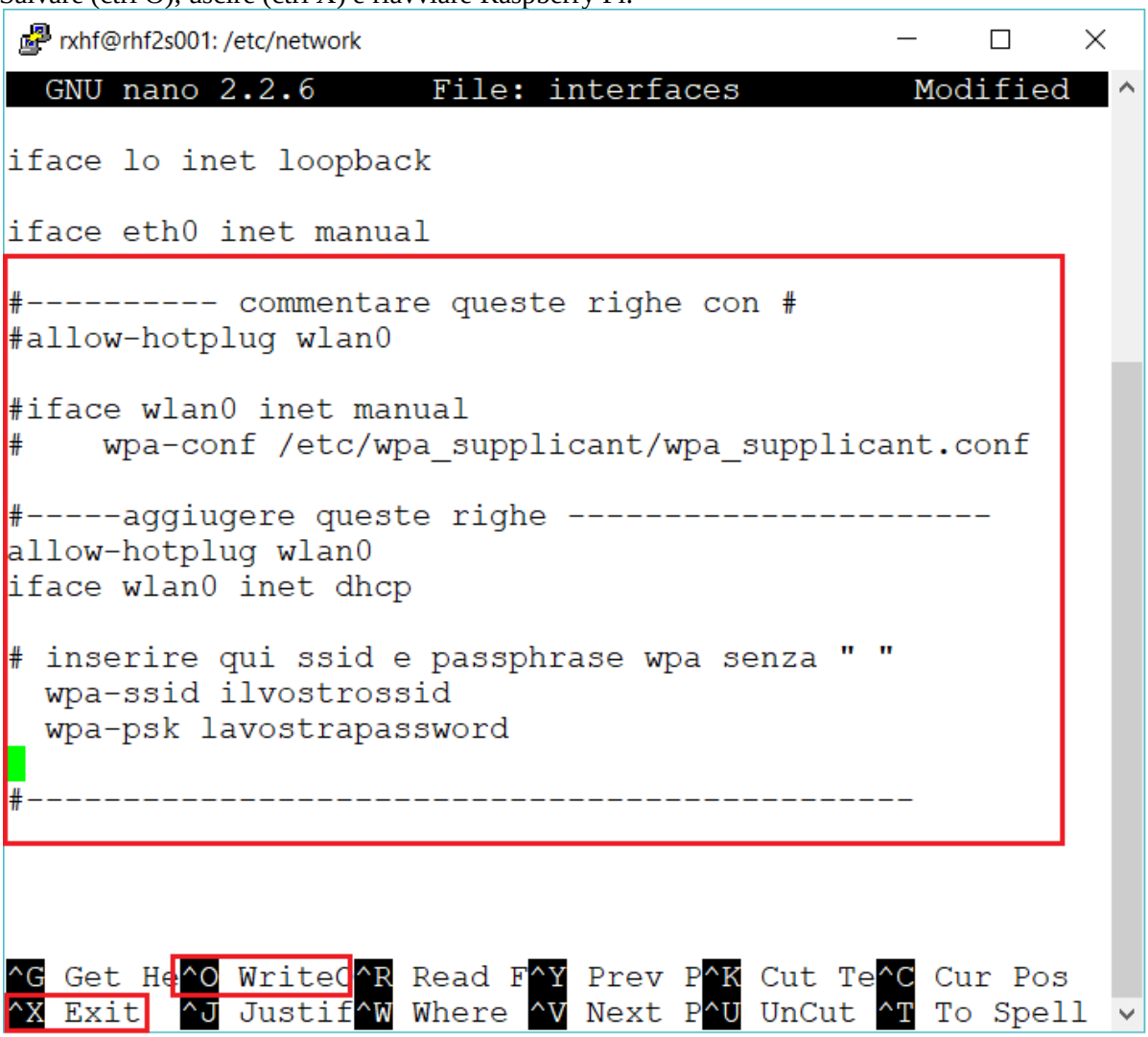


FIGURA 7 Configurazione per accesso in rete via Wi-Fi.

6.2 Configurazione iniziale dell'end node Seeeduino LoRaWAN

Per la configurazione iniziale di Seeeduino LoRaWAN si può procedere nel seguente modo.

1. Aprire il software Arduino IDE e installare la scheda *Seeeduino* LoRaWAN seguendo le istruzioni date dal sito

   http://wiki.seeedstudio.com/Seeed_Arduino_Boards/

   che consistono nel selezionare *File* → *Impostazioni* ed inserire alla voce *Url aggiuntive* per il gestore schede il link indicato (FIGURA 8):

`https://raw.githubusercontent.com/Seeed-Studio/Seeed_Platform/master/package_seeeduino_boards_index.json`

[5] Con versioni diverse del sistema operativo è possibile che si debba operare in altro modo.

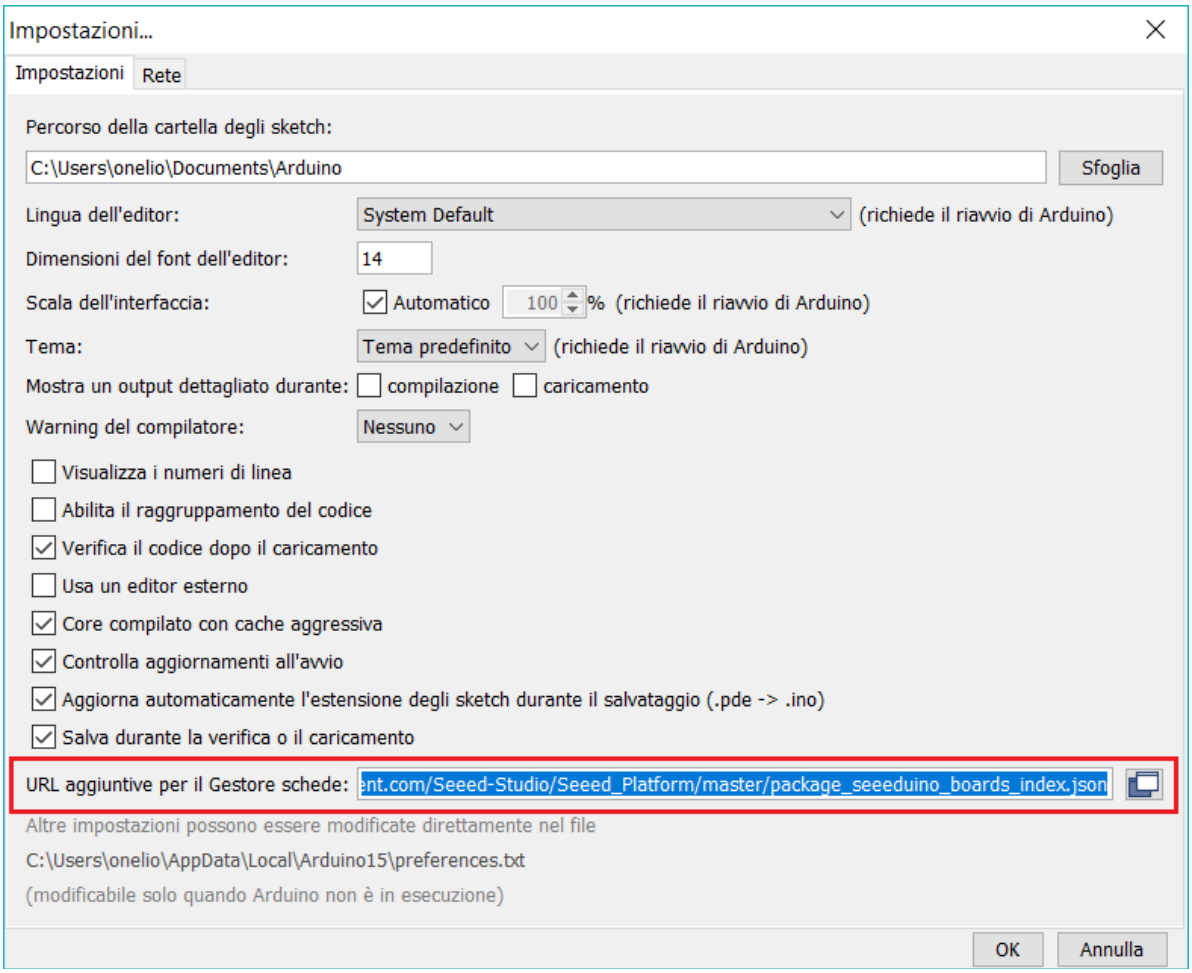


FIGURA 8 Inserimento del link per scaricare il pacchetto per la scheda aggiuntiva Seeeduino.

2. Andare su *Strumenti* → *Scheda* → *Gestore schede* e installare la scheda che supporta *Seeeduino LoRaWAN* (FIGURA 9). Il pacchetto software che si installa comprende anche la libreria **`LoRaWan.h`** che fornisce i metodi da utilizzare negli sketch per trasmettere via LoRa.

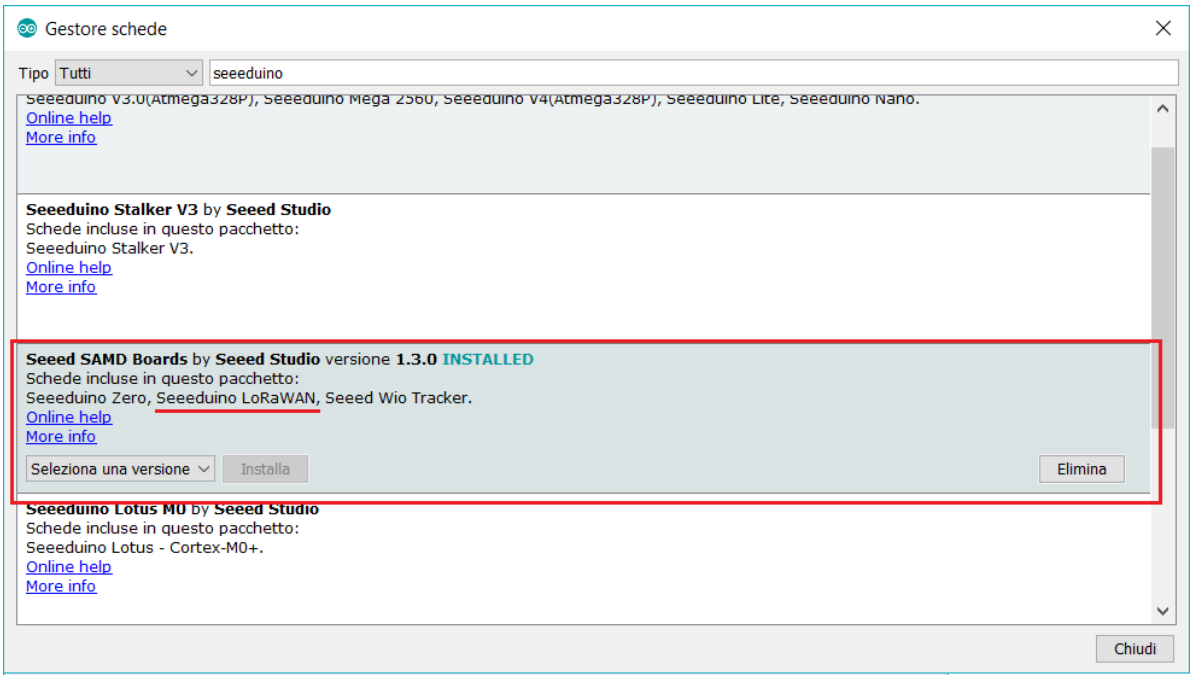


FIGURA 9 Installazione scheda Seeeduino LoRaWAN.

3. Selezionare *Strumenti* → *Scheda* → *Seeeduino LoRaWAN* e la porta su cui è installata la scheda (FIGURA 10).

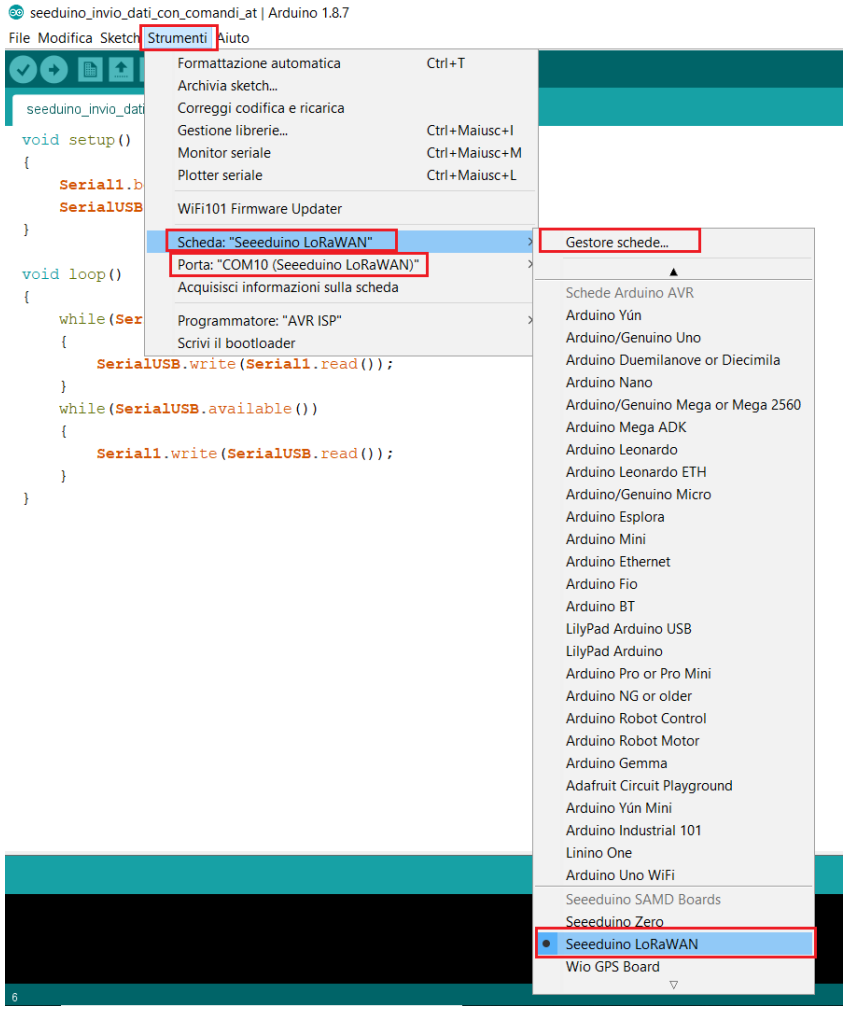


FIGURA 10 Selezione della scheda Seeeduino LoRaWAN e della porta di comunicazione.

4. Utilizzare il seguente sketch per inviare comandi AT a Seeeduino, via monitor seriale, per verificare la sua configurazione iniziale e per le impostazioni generali, configurando il monitor seriale con *Baud rate 9600* e *A capo (NL).*

```
void setup()
{
    Serial1.begin(9600);
    SerialUSB.begin(115200);
}

void loop()
{
    while(Serial1.available())
    {
        SerialUSB.write(Serial1.read());
    }
    while(SerialUSB.available())
    {
        Serial1.write(SerialUSB.read());
    }
}
```

- Ripristinare la configurazione di fabbrica inviando il comando: `AT+FDEFAULT=RISINGHF`

- Caricare le configurazioni del data rate previste per l'Europa nella banda 868 MHz con il comando: `AT+DR=EU868`

- Verificare gli identificativi settati con la configurazione di fabbrica con il comando (FIGURA 11): `AT+ID`

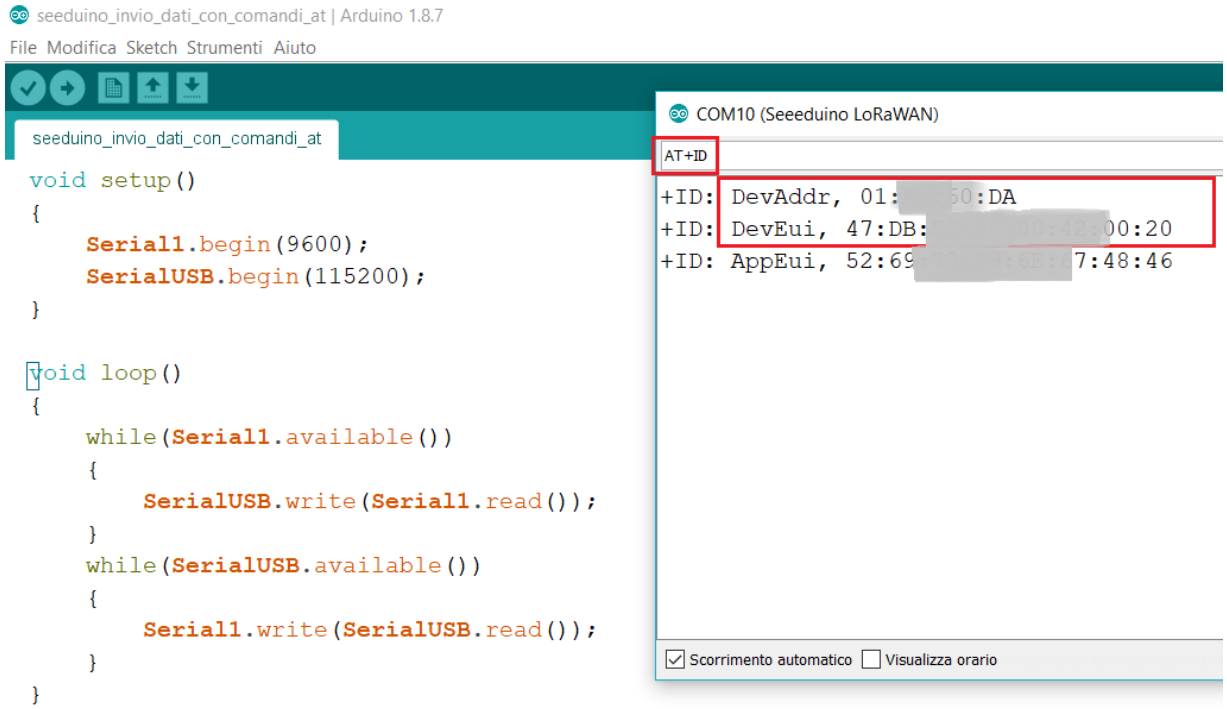


FIGURA 11 Verifica degli identificativi di default di Seeeduino LoRaWAN.

5. Per una prima prova è possibile utilizzare il *network server* integrato in Raspberry Pi (FIGURA 12): si rileva l'indirizzo IPv4 assegnato a Raspberry Pi e da un browser si digita l'indirizzo stesso (nell'esempio il 10.0.0.74)

Si clicca quindi su *Application* e si inseriscono nei campi vuoti le seguenti informazioni:
- Nome dell'applicazione LoRaWAN (nell'esempio `Lab_tele-IoT`);
- Proprietario (*Owner*)
- *AppEui* visualizzata (FIGURA 12) con i comandi AT.

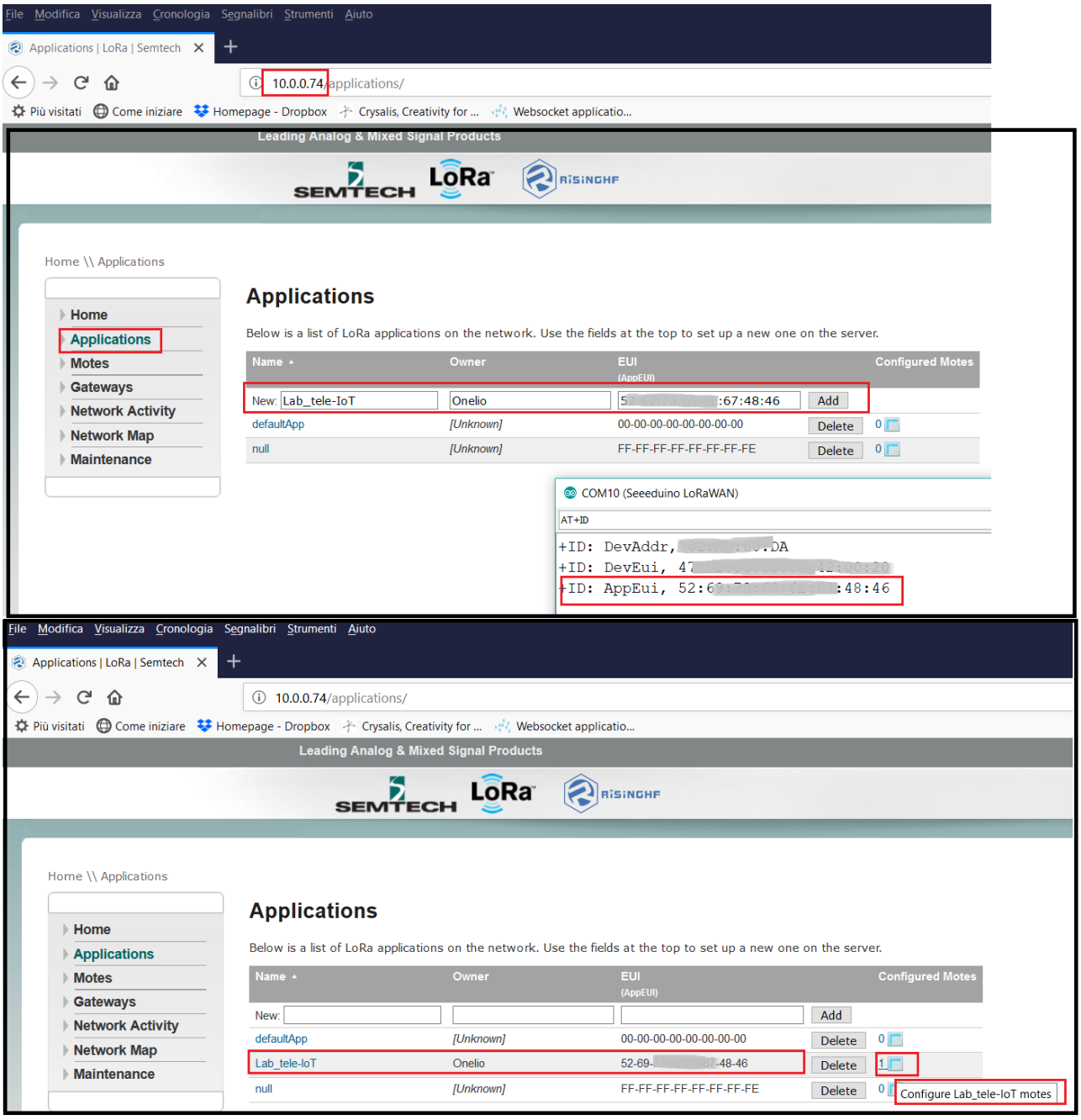


FIGURA 12 Network server integrato nel gateway Raspberry Pi

Si clicca su *Add* e quindi sull'icona sulla destra (*Configure Motes*).
Nella sezione *Personalized Motes* (Modalità ABP, *Activation By Personalization*) nella riga *New* si inseriscono (FIGURA 13):

- gli identificativi *devEui*, *devAddress* visualizzati sul monitor seriale tramite i comandi AT;
- le chiavi *AppSKey* e *NwkSKey* di default (da utilizzare solo per i test iniziali), che sono uguali e date da una stringa fornita nella documentazione del kit (`2B7E15.....CF4F3C`)

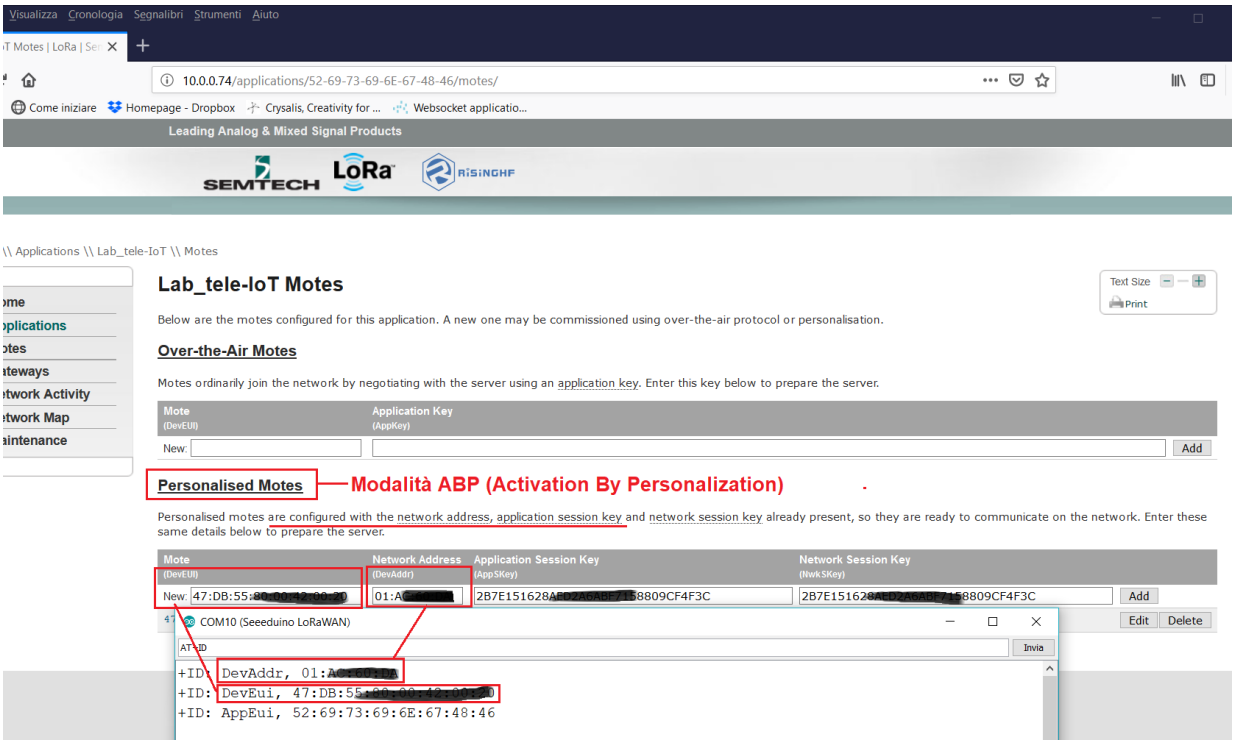


FIGURA 13 Configurazione della modalità ABP sul network server

A questo punto è possibile testare la connettività tra end node Seeeduino LoRaWAN e gateway inviando, tramite i comandi AT indicati, una stringa esadecimale (FIGURA 14):

```
AT+MODE=LWABP
AT+CMSGHEX="0a 0b 0c 0d 0e"
```

selezionando quindi *Network Activity* è possibile ottenere una statistica dei frame ricevuti e la sequenza dei dati inviati da Seeeduino a riprova della corretta connessione con il gateway (Raspberry Pi + modulo LoRa).

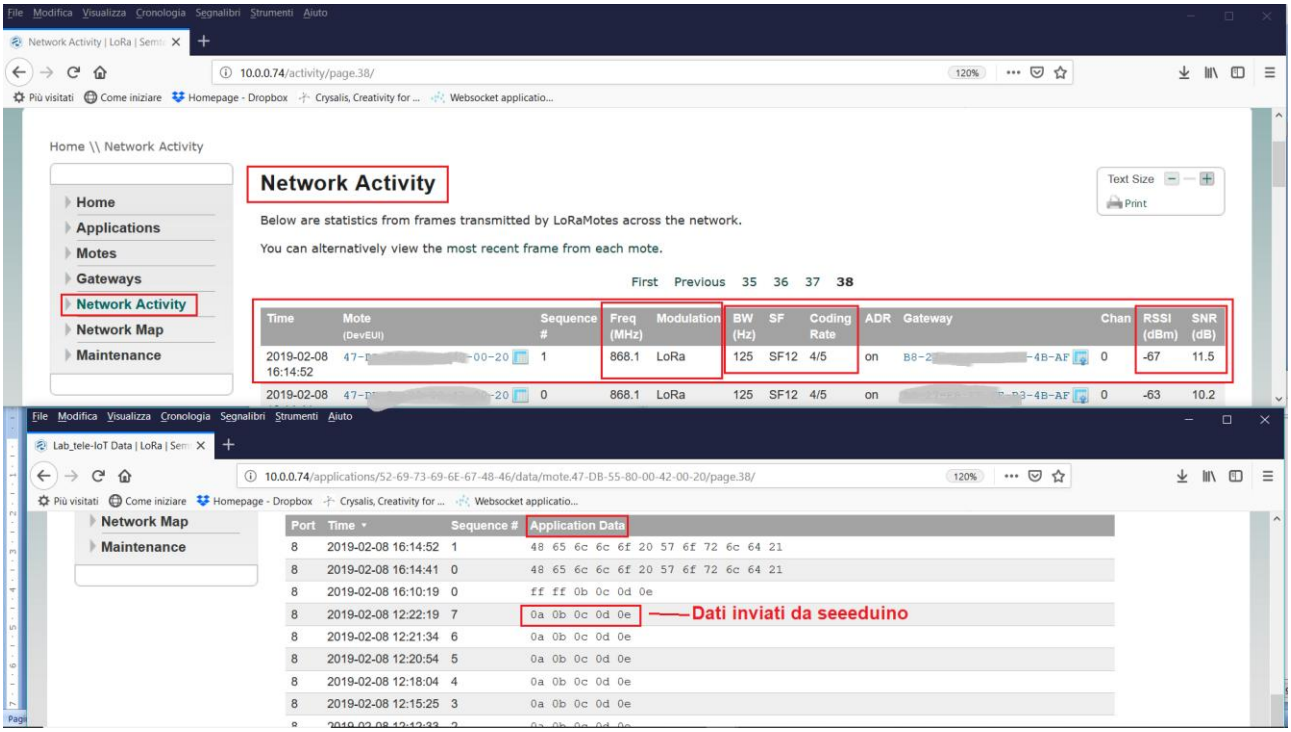


FIGURA 14 Test di connettività con il gateway.

Dopo aver arrestato il network server interno e aver configurato il gateway per la connessione con il network server LORIOT (https://eu1.loriot.io), operando come indicato nel paragrafo 6.5, è possibile effettuare un primo test di connettività (FIGURA 15) per inviare un messaggio testuale utilizzando lo sketch di Seeeduino che consente di inviare comandi AT, digitando sul monitor seriale il comando:

```
AT+CMSG "TESTODELMESSAGGIO"
```

Il messaggio può essere visualizzato tenendo presente che il network server mostra quanto ricevuto in formato esadecimale per cui i dati vanno opportunamente decodificati, cliccando su *Decode data from device* e inserendo il codice che effettua la decodifica (FIGURA 15).

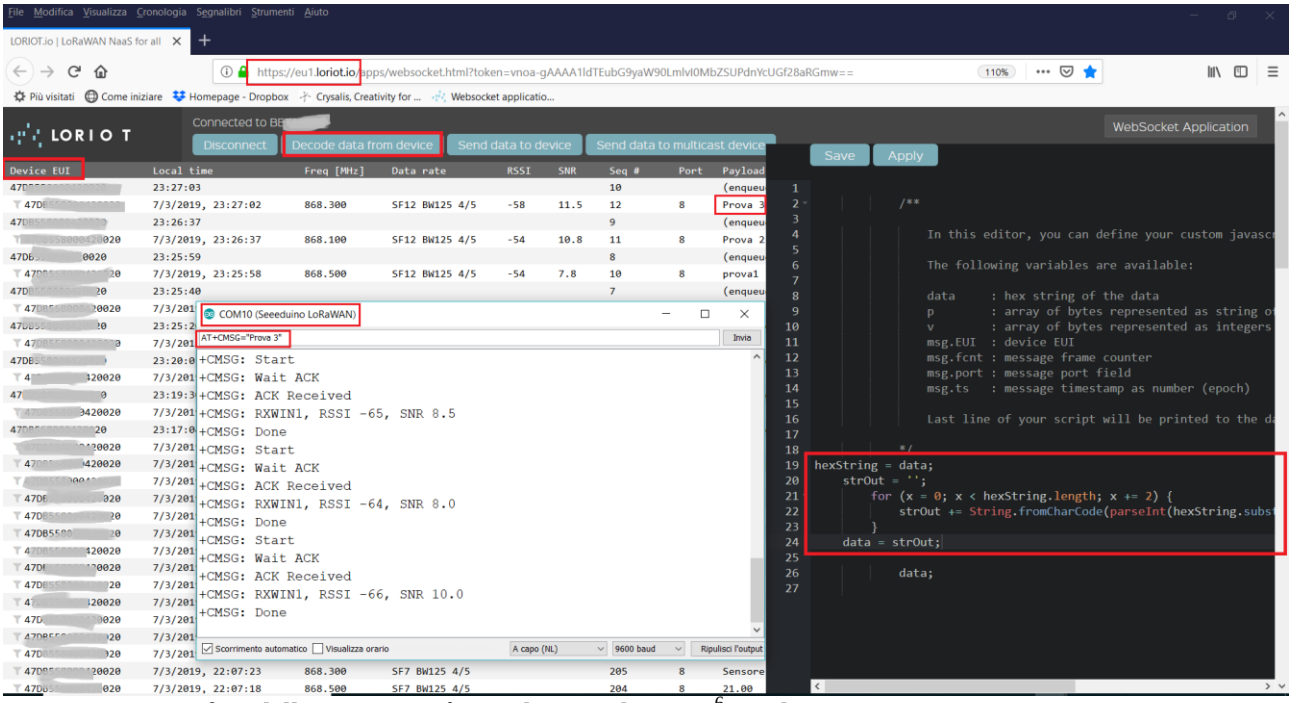


FIGURA 15 Verifica della connettività con il network server[6] eu1.loriot.io

6.4 Configurazione di Arduino MKR WAN 1300

La configurazione dell'IDE di Arduino per la programmazione di un end node *Arduino MKR WAN 1300* è più semplice in quanto è possibile installare il pacchetto per la scheda cliccando su *Strumenti → Scheda → Gestore schede*, cercando e installando il pacchetto per le schede *Arduino SAMD Board* (FIGURA 16).
Si seleziona quindi tra le schede disponibili la scheda Arduino MKR WAN 1300 e la relativa porta di comunicazione.
Si scarica dal sito www.arduinolibraries.info/libraries/mkrwan e si installa la libreria **MKRWAN** che inserita negli sketch mette a disposizione i metodi per la comunicazione via LoRa di Arduino MKR WAN 1300.
Cliccando su *File → Esempi → MKRWAN* si hanno a disposizione degli sketch che esemplificano l'impiego della libreria MKRWAN per la comunicazione via LoRa e per la comunicazione con un network server LoRaWAN. Per ulteriori informazioni si rimanda al sito ufficiale di Arduino.

[6] A seconda della versione del Network Server Loriot, il codice per la decodifica può essere diverso da quello mostrato in figura.

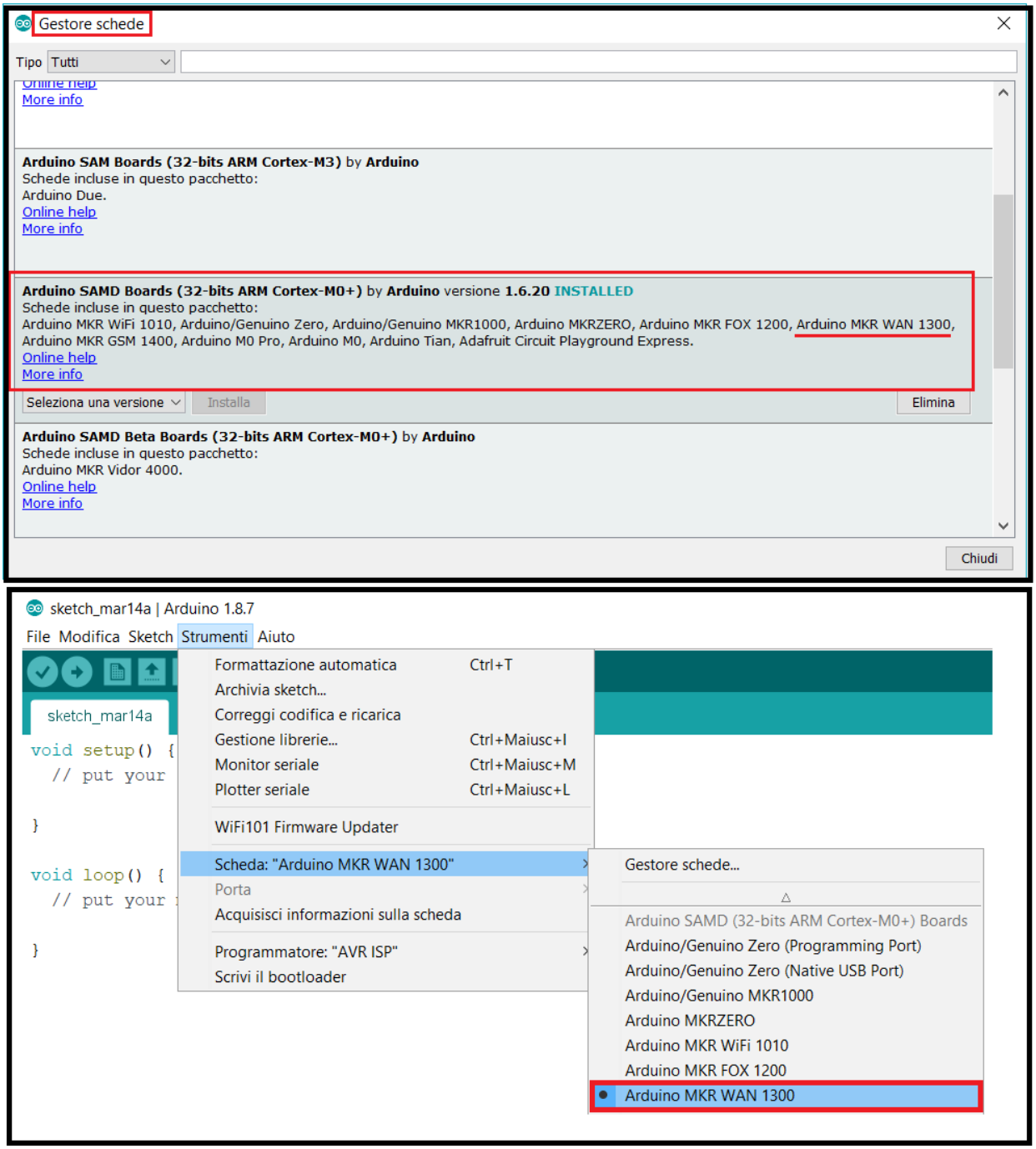


FIGURA 16 Installazione del pacchetto per la scheda Arduino MKR WAN 1300 e selezione della scheda.

6.5 Invio di dati al Network Server in cloud eu1.loriot.io

La piattaforma cloud *LORIOT*, accessibile all'indirizzo www.loriot.io, tra l'altro mette a disposizione dei propri utenti un *Network Server* LoRaWAN, un'applicazione per visualizzare i dati raccolti e la possibilità di esportare i dati stessi verso altre piattaforme cloud su Internet in diversi modi.
L'accesso alla piattaforma è gratuito per una *network application* LoRaWAN che comprende 1 gateway e fino a 10 *end node* (detti anche *device* o *mote*), che nel nostro caso vanno attivati (registrati, *Enroll*) in modalità *ABP* (*Activation By Personalization*).
I passi da effettuare per consentire la ricezione dei dati inviati dagli end node (device) da parte del network server LORIOT (eu1.loriot.io) sono i seguenti.

1. Con un browser ci si collega al sito www.loriot.io , si clicca su *LOG IN*, si seleziona il *Network Server* più vicino, nel nostro caso il server EU1, situato a Francoforte in Germania, accessibile direttamente al link https://eu1.loriot.io , si clicca su *Register a new account* e si inseriscono i dati richiesti.

2. Si effettua il *Log In* al sito https://eu1.loriot.io e si accede alla *dashboard* tramite cui si configurano e si attivano il *gateway* e gli *end node*.

3. Si clicca su *Register a new gateway* (FIGURA 17) per registrare il proprio gateway, si seleziona tra tutti i gateway disponibili *Raspberry Pi 3* (FIGURA 18); si inserisce l'indirizzo MAC del gateway (individuabile seguendo le istruzioni), il radio *Front end* RHF2S001 868/915 MHz (SX1257) e la sua localizzazione, compilando il form presentato; si clicca infine su *Register Raspberry Pi 3 gateway*.

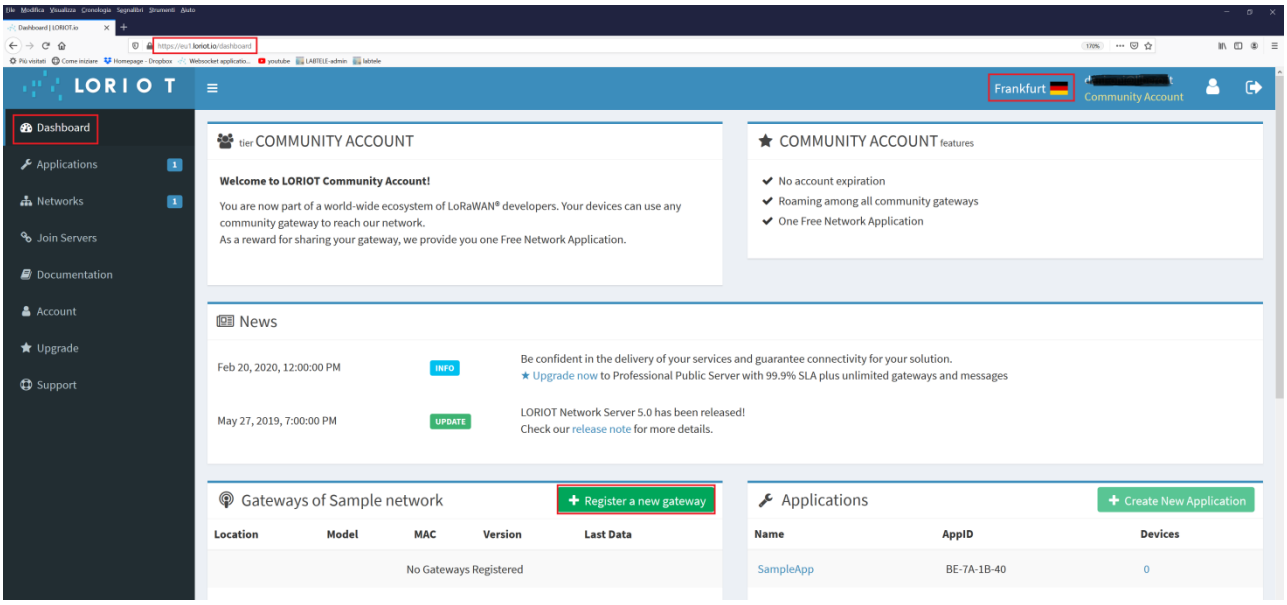


FIGURA 17 Registrazione del gateway.

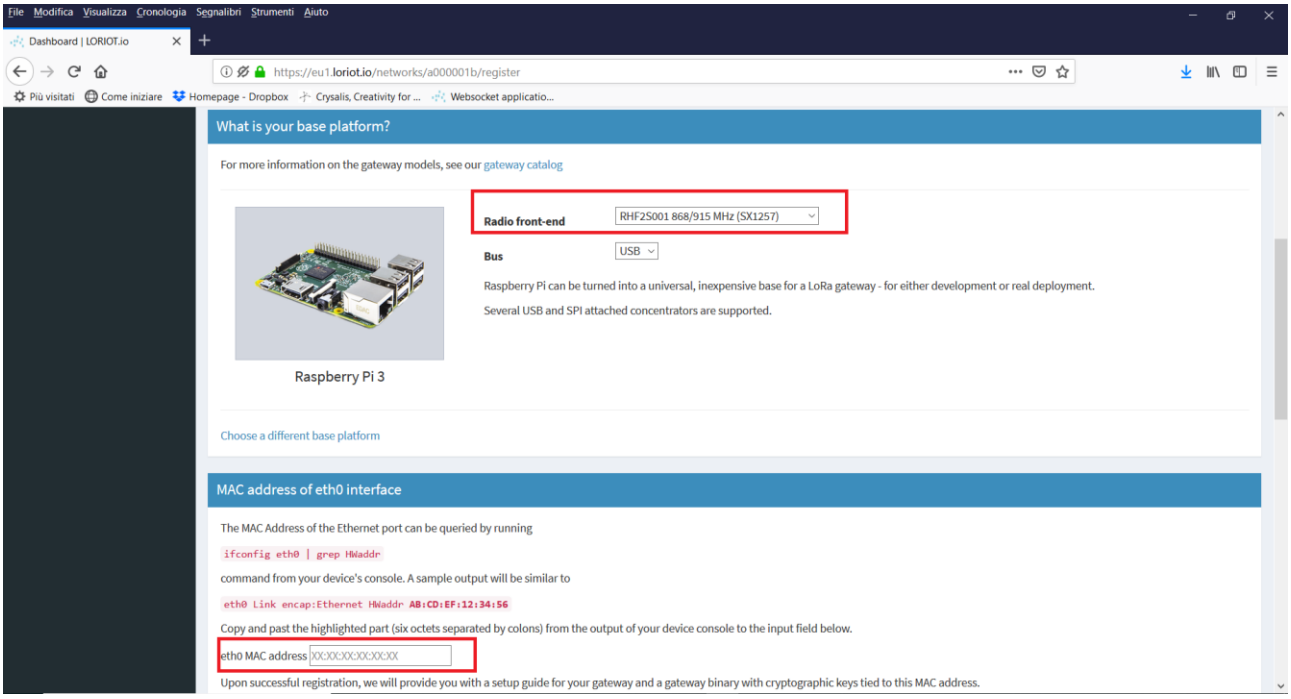


FIGURA 18 Selezione del tipo di gateway.

4. Si torna alla dashboard e cliccando su *Applications* è possibile modificare il nome di default *SampleApp* in uno più significativo, per esempio LAB_TELE_IOT o IOT_LAB_SENSORI.
5. Si clicca su *Applications* → <*nome Application*> e si seleziona *Devices* per registrare e attivare un *end device* cliccando su *Enroll Device*. Per semplificare la configurazione degli end device si è scelto di utilizzare la versione **LoRAWAN 1.0.x** (FIGURA 19) che richiede la configurazione di un numero inferiore di parametri rispetto alle versioni successive.

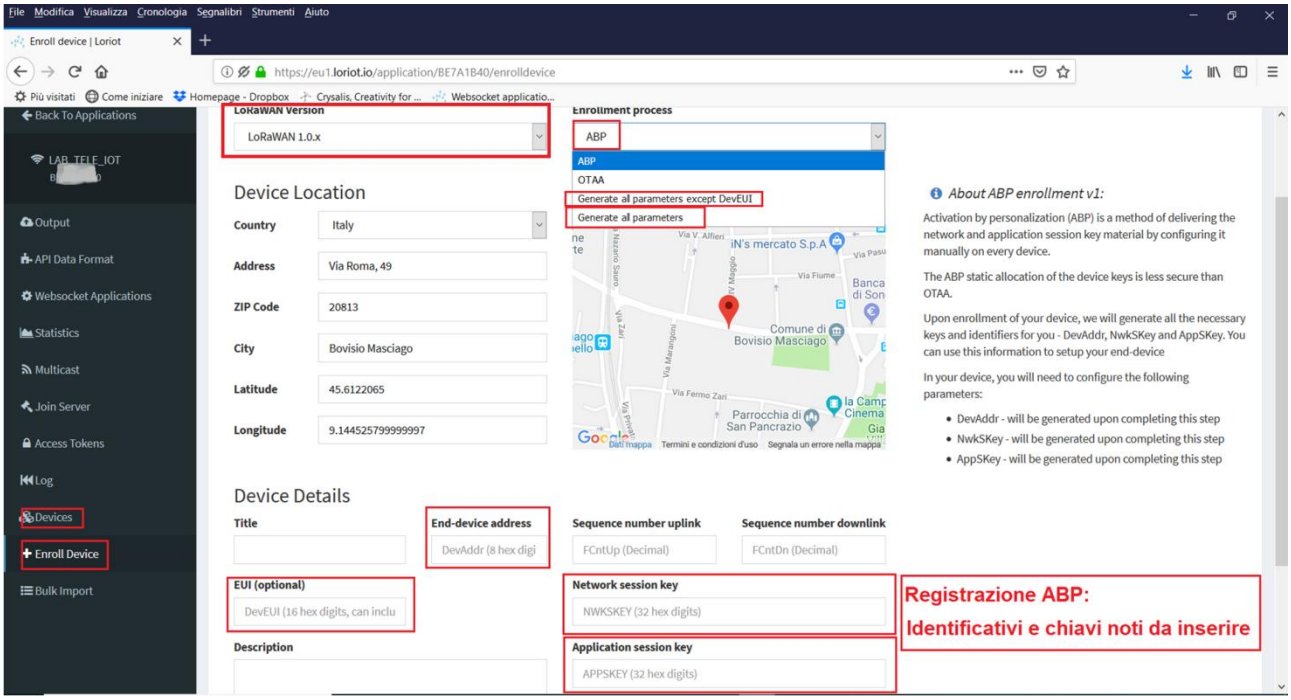


FIGURA 19 Registrazione con la versione LoRAWAN 1.0.x e la modalità ABP (*Activation By Personalization*) senza generazione di parametri.

6. Si seleziona il processo di registrazione (*Enrollment Process*) scegliendo tra:
   - *ABP*, se si conoscono già tutti i parametri necessari per la registrazione, che sono *DevEui*, *DevAddress*, *NetworkSessionKey (NwkSKey) ApplicationSessionKey (AppSKey)*, FIGURA 19; è questo il caso in cui si registra *Seeeduino LoRaWAN* per un primo test, utilizzando gli identificativi preimpostati in fabbrica e le chiavi di default; si clicca quindi su *Enroll* per registrare l'end node;
   - *Generate all parameters except DevEui*; è questo il caso in cui si desidera mantenere l'EUI (*Extended Unique Identifier*) dell'end node, nel nostro caso un *Arduino MKR WAN 1300*, ma non si conoscono *DevAddr*, *NwkSKey* e *AppSKey* che vengono fatti generare dal Network Server LORIOT e sono ottenibili cliccando sul *device* appena registrato (FIGURA 20);
   - *Generate all parameters* (FIGURA 21); con cui vengono generati tutti i parametri, compreso il *DevEui*; sarà poi necessario configurare sull'end node (per esempio Seeeduino LoRaWAN) il nuovo *DevEui* ed il nuovo *DevAddr* tramite l'invio di appositi comandi AT, impiegando lo sketch che consente di fare ciò. E' consigliabile modificare i parametri di accesso e non utilizzare quelli di default (tranne eventualmente il DevEui).

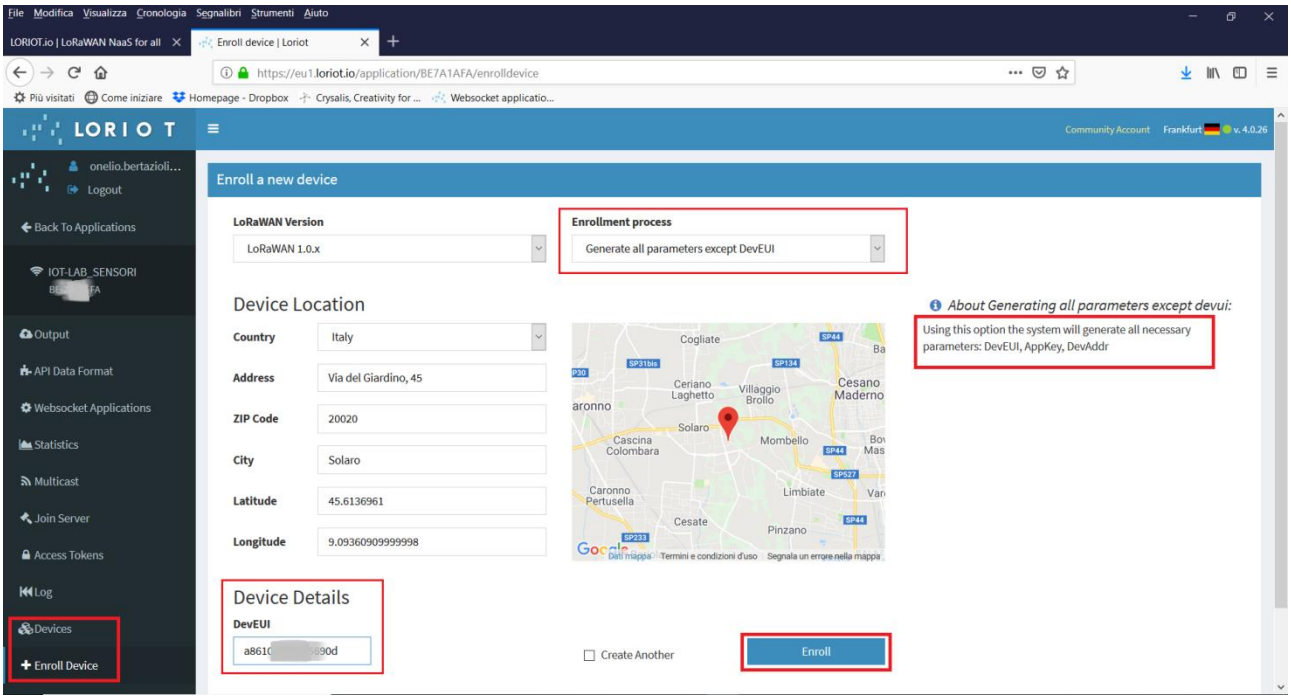


FIGURA 20 Registrazione in modalità ABP con generazione di tutti i parametri tranne DevEUI

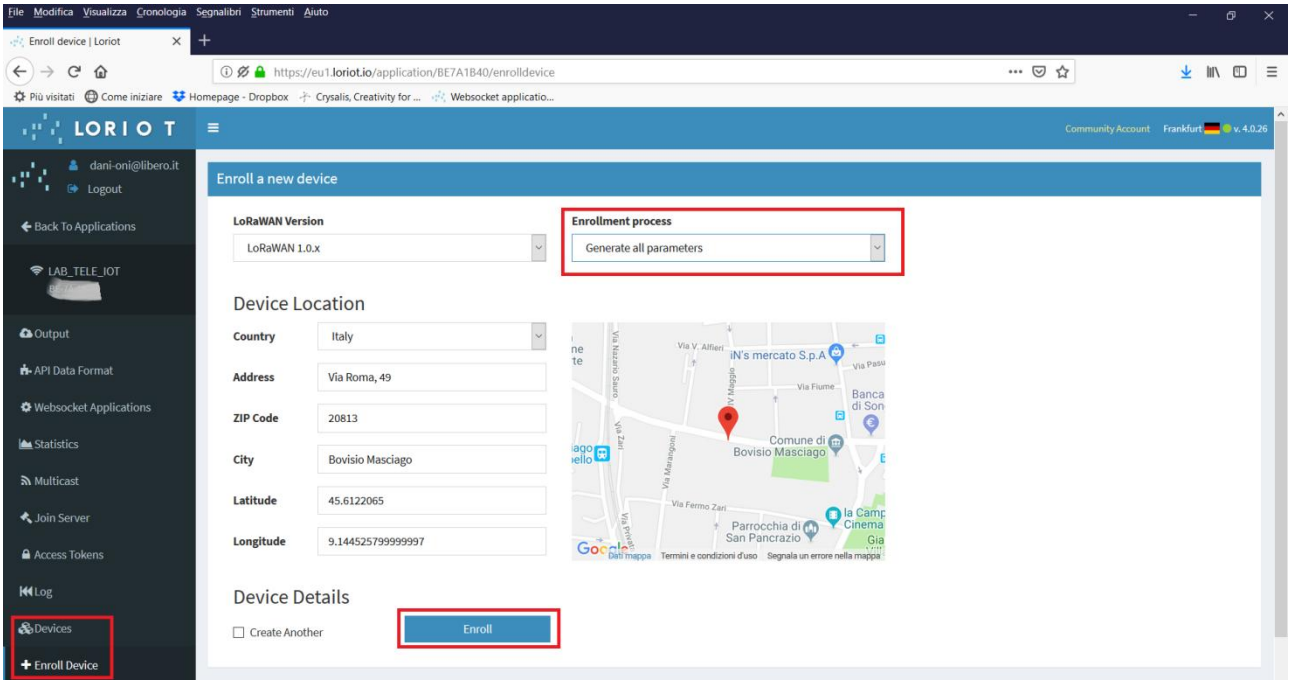


FIGURA 21 Registrazione in modalità ABP con generazione di tutti i parametri.

7. Una volta registrato il *device* (*end node*), cliccando su esso (FIGURA 22) è possibile visualizzare tutti i parametri di accesso, selezionando dalla pagina dei *Devices* il dispositivo desiderato; le chiavi sono visualizzabili cliccando su *LoRaWAN Parameters* e possono essere inserite nello sketch con cui si inviano i dati al network server (tramite il gateway).

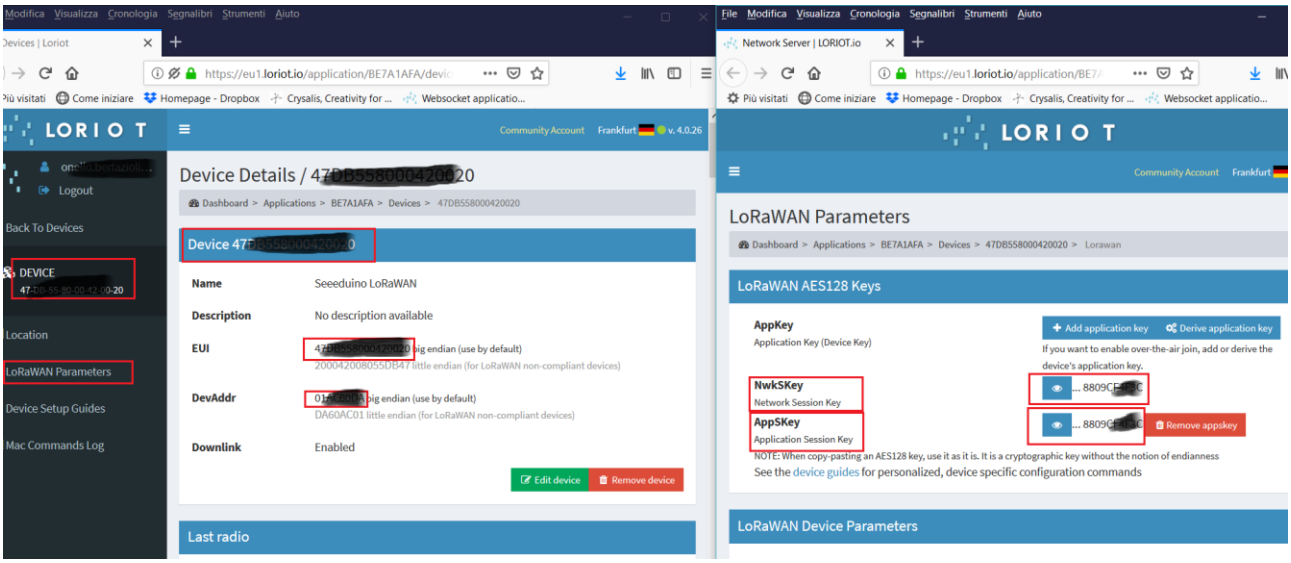


FIGURA 22 Visualizzazione degli identificativi e delle chiavi per un *device*.

8. Cliccando su *Applications* → <*nomeApplication*> → *Websocket Applications* → *Websocket Sample* (FIGURA 23) è possibile visualizzare in tempo reale i dati inviati dagli *end node* (tramite il gateway), con i relativi parametri. I dati vengono visualizzati in formato esadecimale (FIGURA 24); è però possibile visualizzarli in formato testo cliccando su *Decode data from device*, inserendo il codice (in JavaScript) per la decodifica; nel caso esemplificato in FIGURA 25 in cui i sensori inviano cinque dati numerici, ciascuno composto da cinque caratteri (es. 55.00 22.00 ….), il codice applicato è stato il seguente (dalle istruzioni fornite dalla piattaforma si sa che "v: array of bytes represented as integer"):

```
var x;
v.toString();
var data = String.fromCharCode(v[0]);
        for (x = 1; x < 25; x++) {
            if( v[x] != ""){
            data = data + String.fromCharCode(v[x]);}
          }
     data;
```

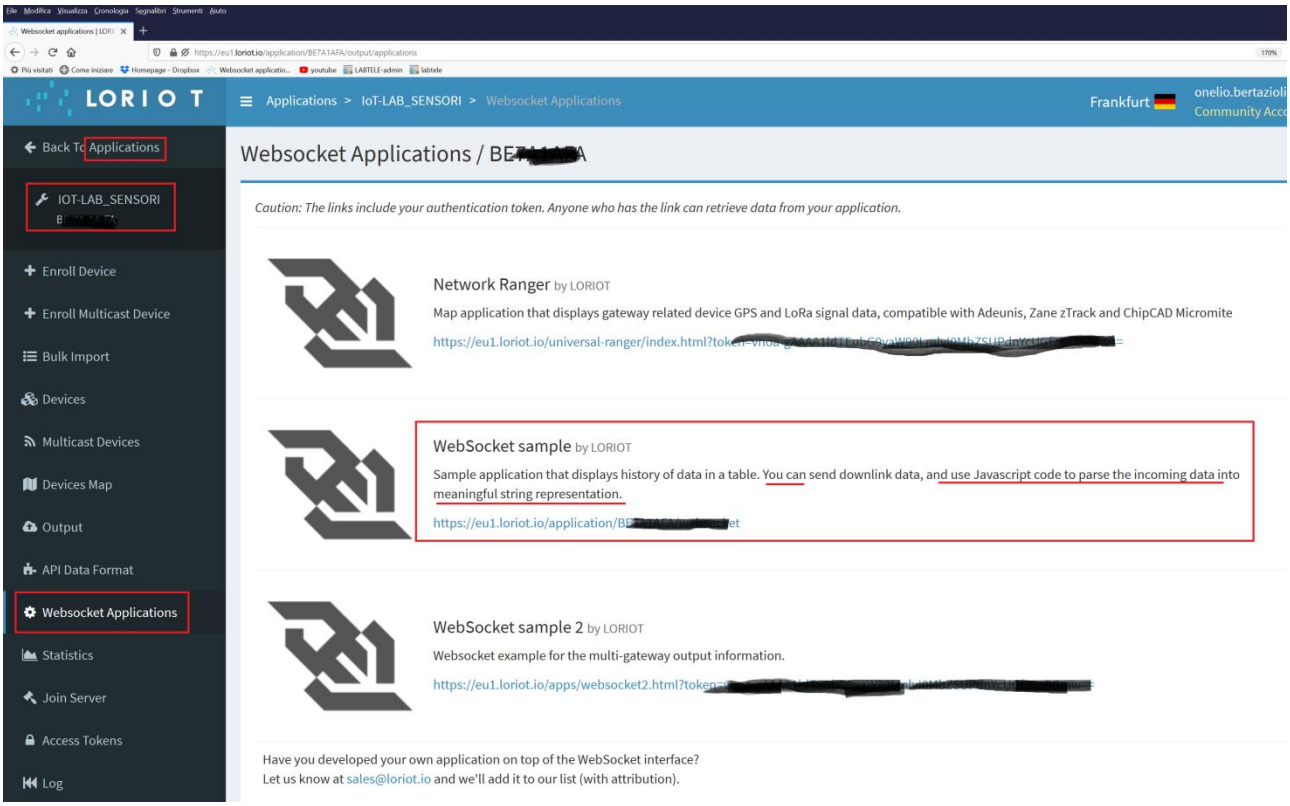


FIGURA 23 Selezione della modalità di visualizzazione dei dati

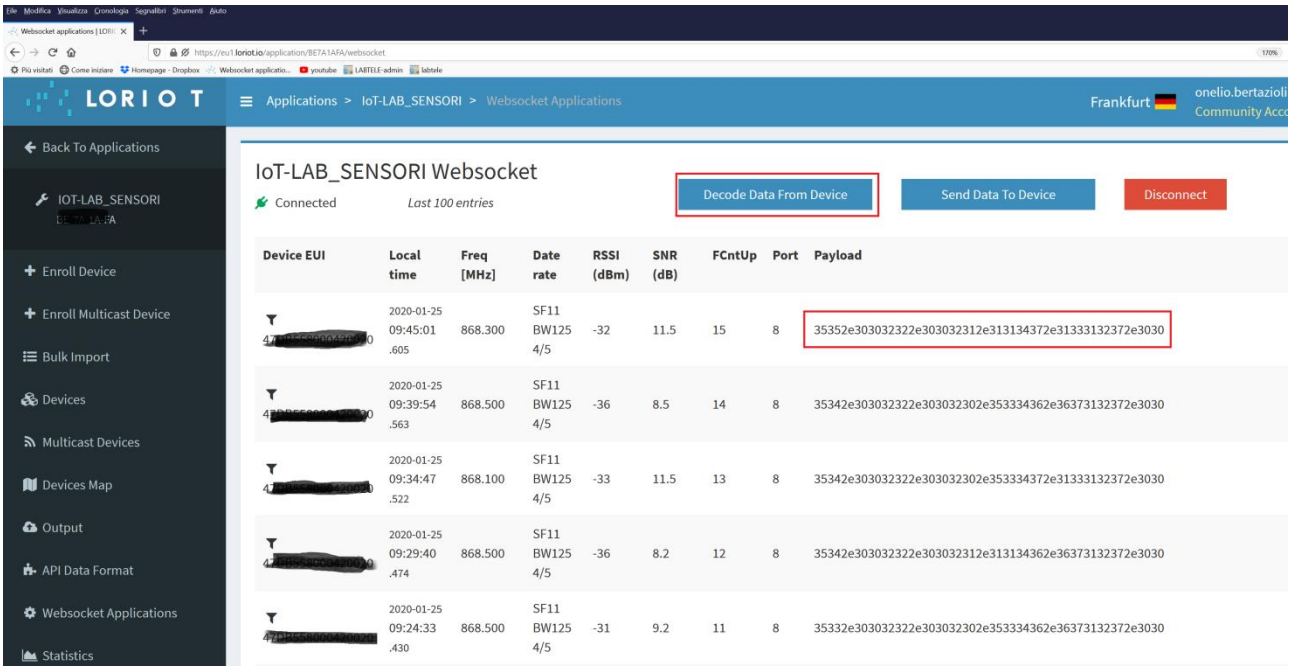
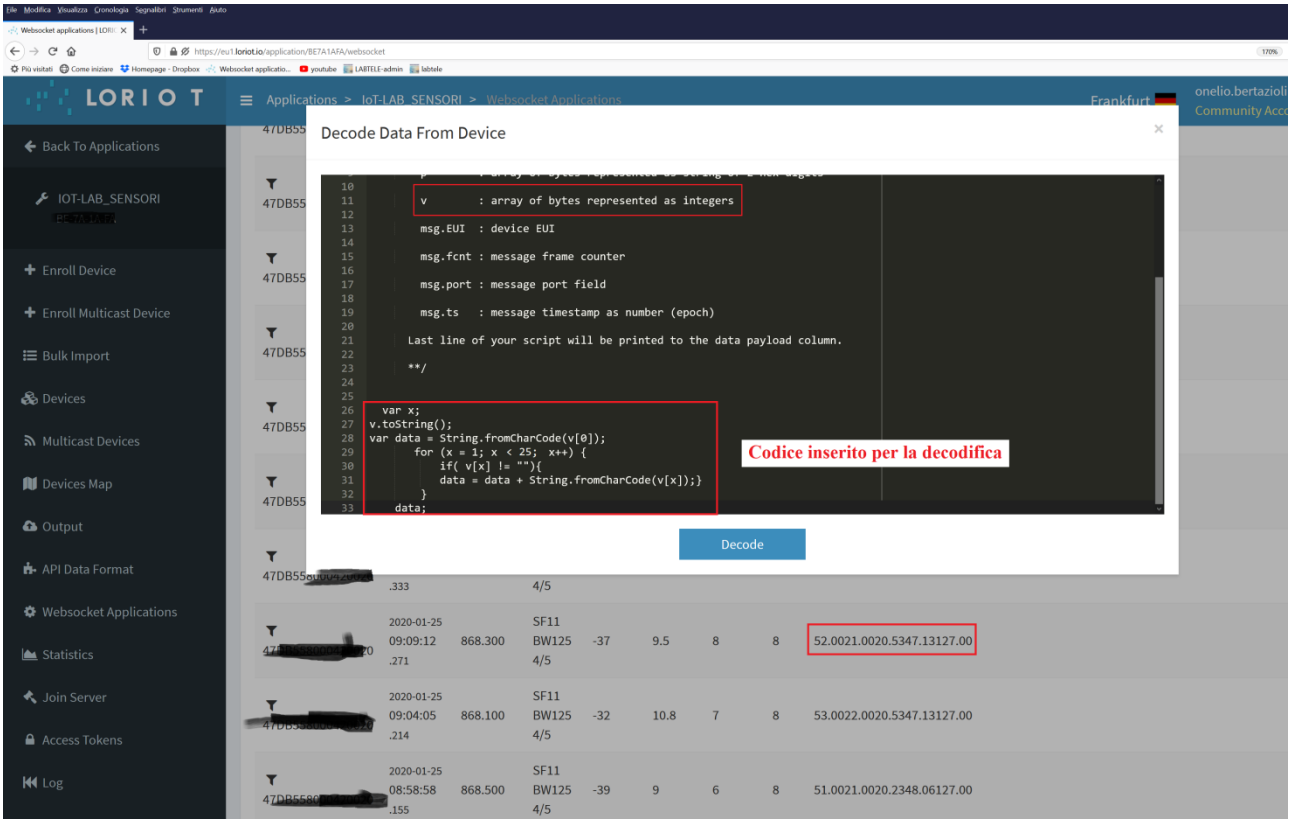


FIGURA 24 Visualizzazione dei dati in formato esadecimale e pulsante per l'inserimento del codice (in JavaScript) per la decodifica.

Hex	Char	Hex	Char	Hex	Char
20	Space	40	@	60	`
21	!	41	A	61	a
22	"	42	B	62	b
23	#	43	C	63	c
24	$	44	D	64	d
25	%	45	E	65	e
26	&	46	F	66	f
27	'	47	G	67	g
28	(	48	H	68	h
29	)	49	I	69	i
2A	*	4A	J	6A	j
2B	+	4B	K	6B	k
2C	´	4C	L	6C	l
2D	–	4D	M	6D	m
2E	.	4E	N	6E	n
2F	/	4F	O	6F	o
30	0	50	P	70	p
31	1	51	Q	71	q
32	2	52	R	72	r
33	3	53	S	73	s
34	4	54	T	74	t
35	5	55	U	75	u
36	6	56	V	76	v
37	7	57	W	77	w
38	8	58	X	78	x
39	9	59	Y	79	y
3A	:	5A	Z	7A	z
3B	;	5B	[	7B	{
3C	<	5C	\	7C	|
3D	=	5D	]	7D	}
3E	>	5E	^	7E	~
3F	?	5F	_	7F	Del

Esempi di corrispondenza tra byte espressi in esadecimale e valori numerici corrispondenti



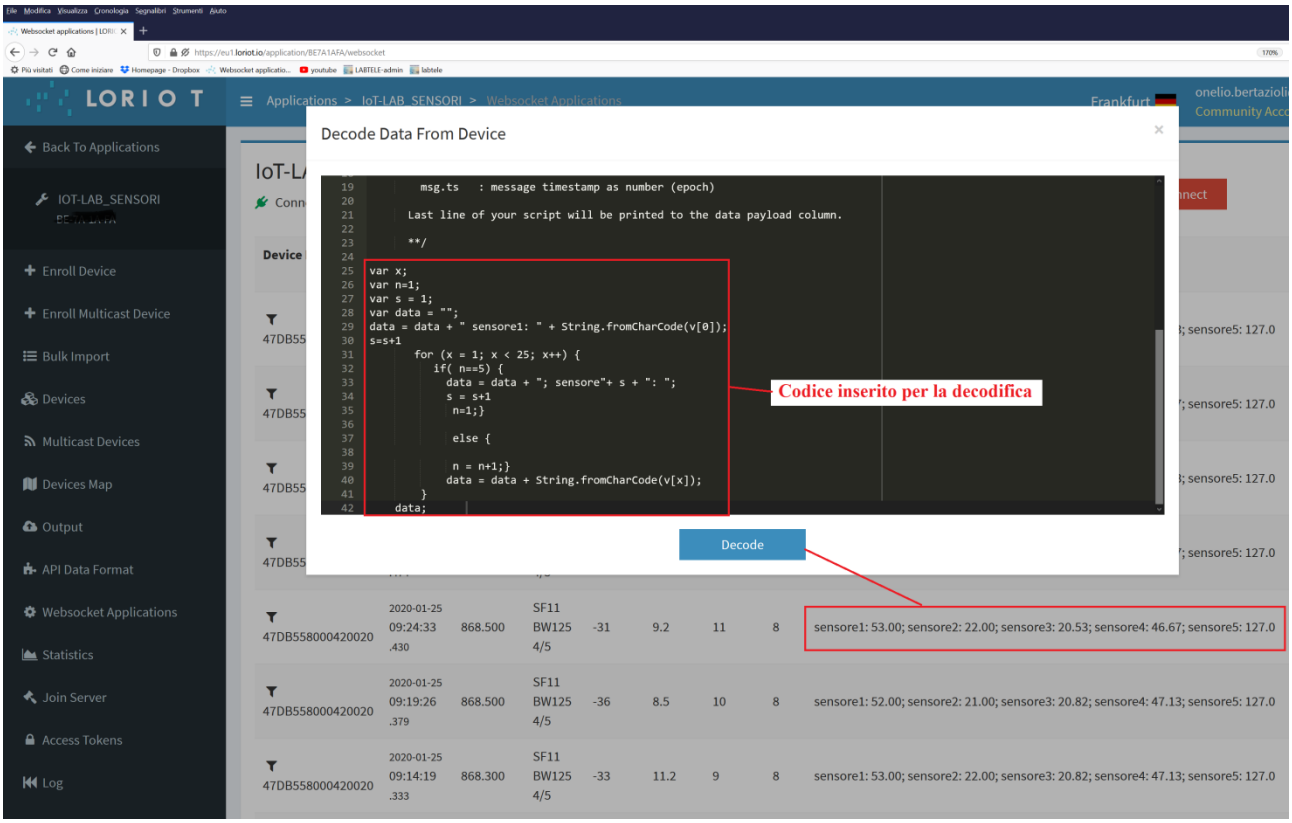


FIGURA 25 Esempi di codice per la decodifica.

Le principali informazioni che vengono visualizzate dal network server sono le seguenti (FIGURA 26):

- identificativo del device (*DevEui*);
- frequenza del canale su cui trasmette;
- *Data rate*, espresso in termini di combinazione di *Spreading Factor* (SF), *Banda di canale* (BW), code rate (CR = 4/5) come indicato dalla FORMULA 1;
- *RSSI* (*Received Strength Signal Indicator*), intensità del livello di potenza totale ricevuto, comprensivo di segnale e rumore;
- *SNR* (*Signal to Noise Ratio*), rapporto Segnale-Rumore; valuta la qualità del segnale ricevuto; LoRa impiega una tecnica di trasmissione *Spread Spectrum* che consente di operare con bassissimi valori di SNR, anche negativi (fino a quasi -20 dB);
- *Payload*, sono i dati veri e propri (o i messaggi) inviati dagli end node (temperatura rilevata dai sensori ecc.).

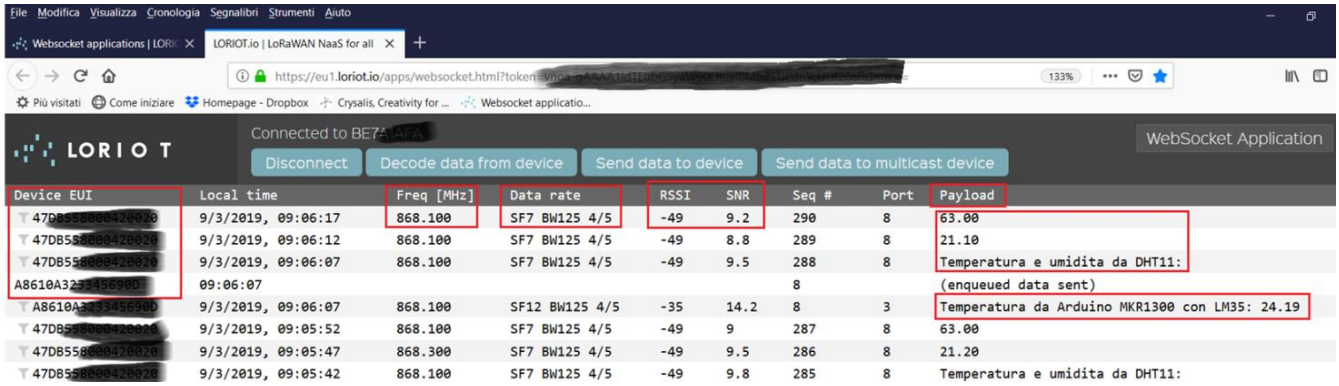


FIGURA 26 Altro esempio di visualizzazione dei dati in formato testo dopo l'applicazione del codice per la decodifica.

6.6 Esportazione dei dati da LORIOT verso altre piattaforme cloud

Per concludere, si accenna la fatto che cliccando su *Applications* → <*nomeApplication*> →*Output* → *Add new output* è possibile esportare i dati (grezzi, in formato esadecimale) verso un'altra piattaforma cloud, come per esempio *AllThingsTalk maker* (https://maker.allthingstalk.com), seguendo le istruzioni riportate (FIGURA 27).
Attraverso quest'ultima piattaforma, applicando le opportune impostazioni e la decodifica (FIGURA 28) è possibile ottenere diversi tipi di visualizzazione dei dati, nonché una mappa che indica la posizione dei sensor node, come esemplificato nelle FIGURE 29 e 30.

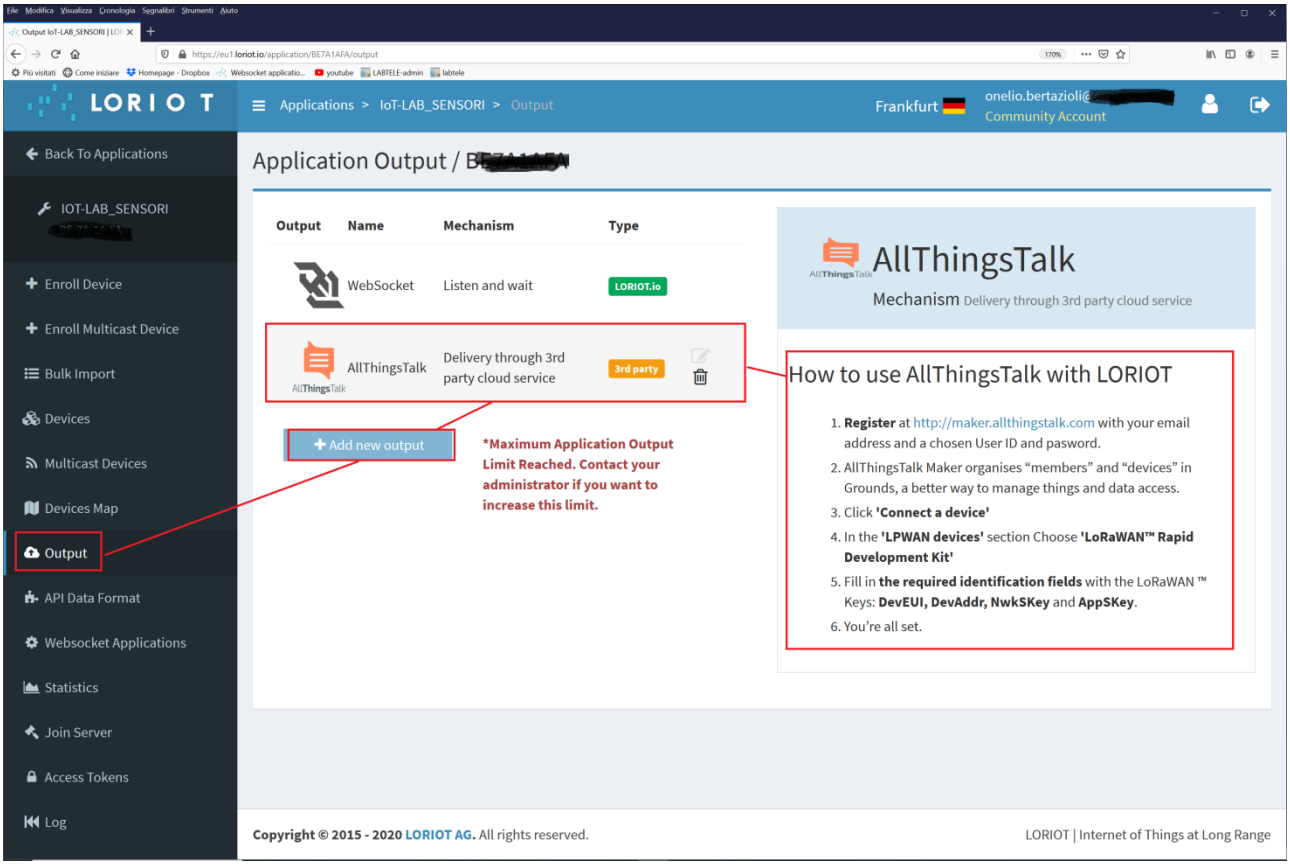


FIGURA 27 Attivazione dell'esportazione (*Output*) dei dati grezzi verso la piattaforma *AllThingsTalk maker*

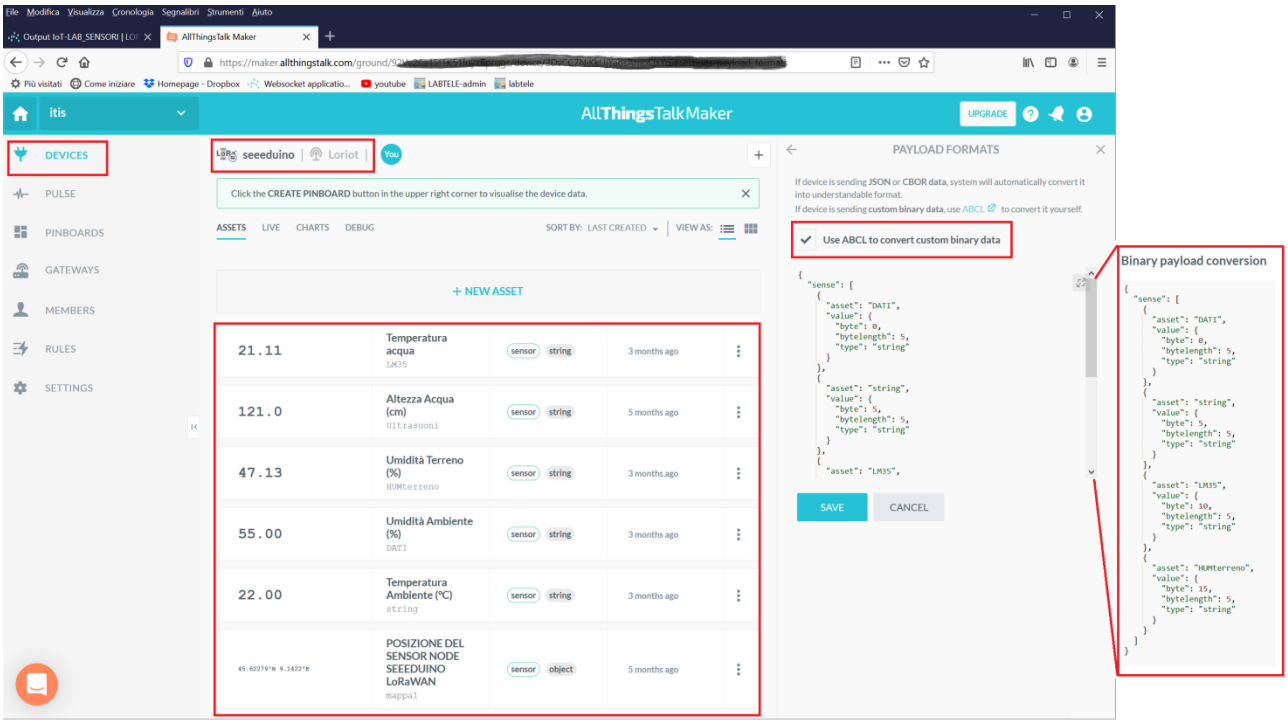


FIGURA 28 Esempio di impostazioni per la visualizzazione dei dati dei sensori montati su Seeeduino LoRaWAN, ricevuti da AllThingsTalk attraverso il network server LORIOT e relativa decodifica.

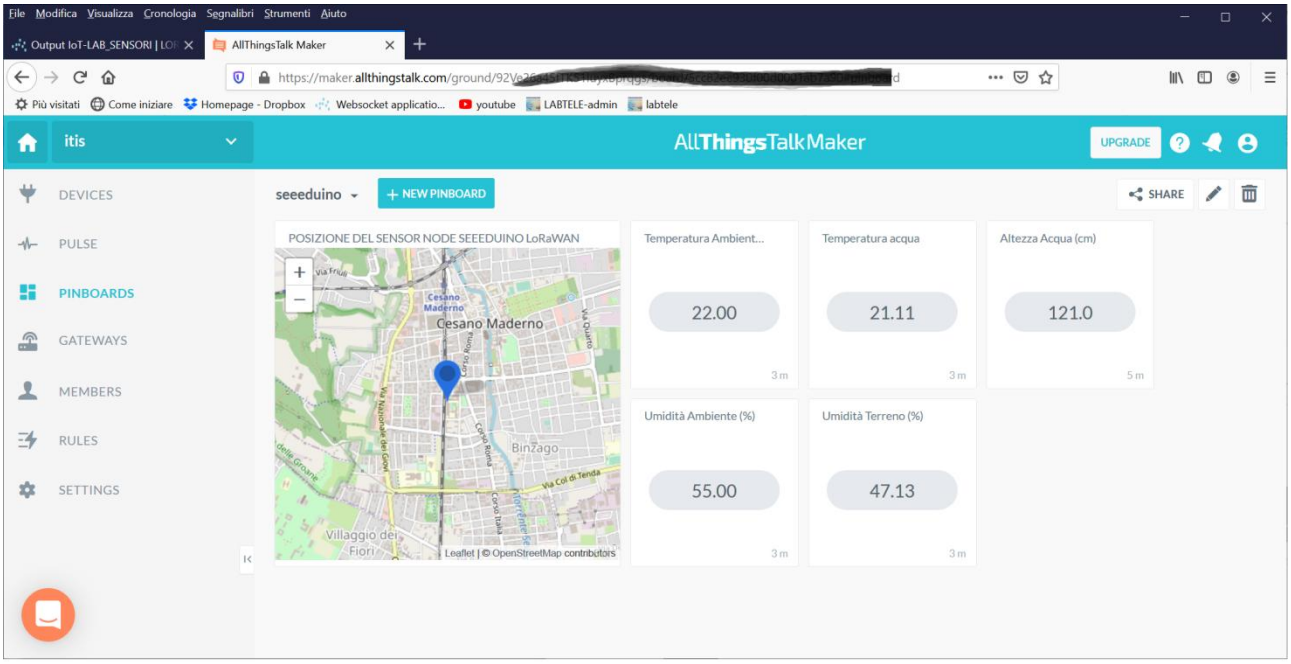


FIGURA 29 Visualizzazione ottenuta con le impostazioni di FIGURA 28

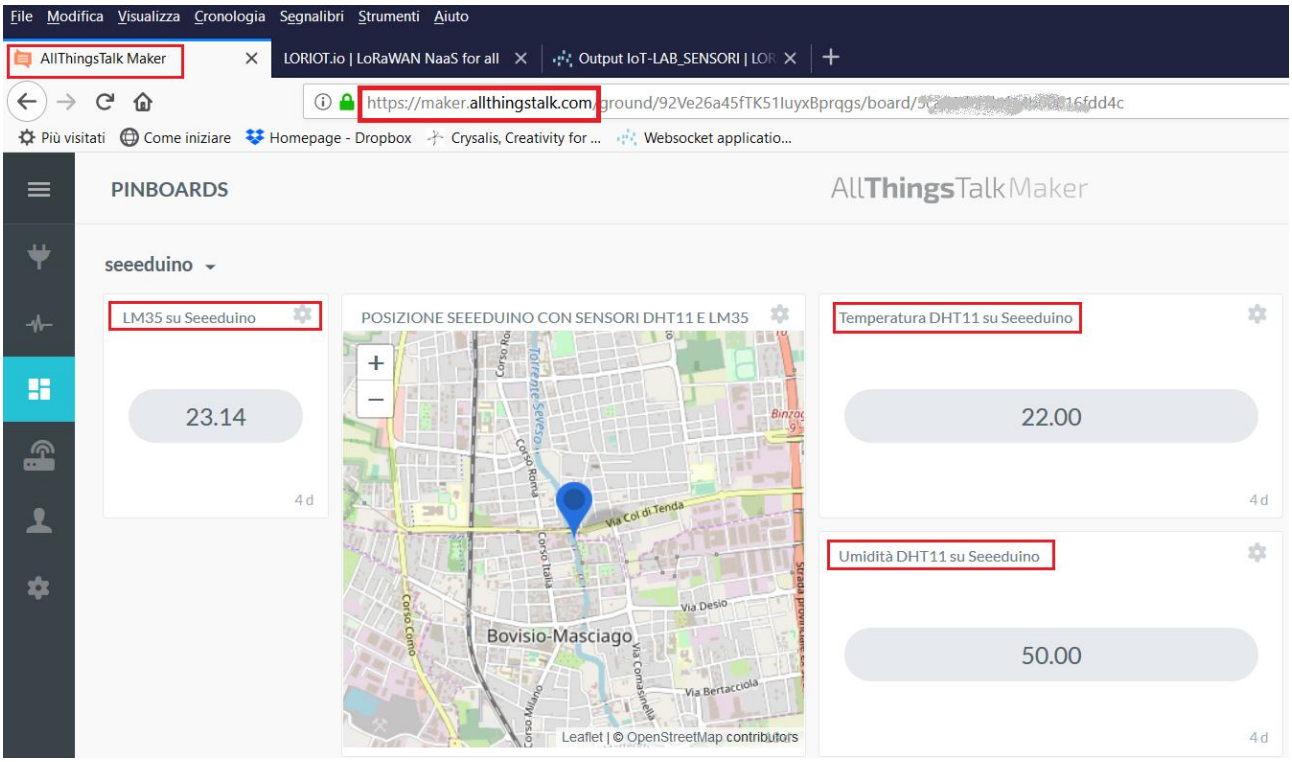


FIGURA 30 Altro esempio di dati visualizzati sulla piattaforma cloud AllThingsTalk maker

Esempi di sketch

Tra le risorse online del libro di testo sono disponibili alcuni sketch per Seeeduino LoRaWAN e per Arduino MKR WAN 1300:

- sketch per la configurazione iniziale di Seeeduino LoRaWAN;
- sketch per Seeeduino LoRaWAN con sensore di temperatura LM35, con invio dati al monitor seriale e al network server LORIOT (https://eu1.loriot.io) tramite il gateway Raspberry Pi;
- sketch per Seeeduino LoRaWAN con sensore di temperatura LM35, con invio dati al network server **eu1.loriot.io** (tramite il gateway Raspberry Pi), senza invio di dati al monitor seriale;
- sketch per Seeeduino LoRaWAN con sensore di temperatura e umidità DHT11, con invio dati al monitor seriale, con invio dei dati al network server eu1.loriot.io (tramite il gateway Raspberry Pi);
- sketch per Seeeduino LoRaWAN alimentato a batteria, sensore di temperatura e umidità DHT11, senza invio dati al monitor seriale, con invio dei dati al network server eu1.loriot.io (tramite il gateway Raspberry Pi);
- sketch per Arduino MKR WAN 1300 con sensore di temperatura LM35, con invio di dati al monitor seriale e al network server eu1.loriot.io (tramite il gateway Raspberry Pi)
- sketch per Arduino MKR WAN 1300 alimentato a batteria, con sensore di temperatura LM35, con invio dei dati al network server eu1.loriot.io (tramite il gateway Raspberry Pi), senza invio di dati al monitor seriale.

Esempio di sketch per Seeeduino LoRaWAN alimentato a batteria, con sensore di temperatura e umidità DHT11, con il solo invio di dati al network server **eu1.loriot.io** tramite il gateway Raspberry PI. Sono state apportate delle modifiche alla libreria LoRaWan, cambiandone anche la denominazione in `LoRaWanmd.h` (FIGURA 31) per poter operare a batteria[7] (la libreria modificata è disponibile tra le risorse online del testo).

```
/* Sketch a corredo del libro di testo
   Onelio Bertazioli Telecomunicazioni - Seconda edizione ed. Zanichelli*/

/* MISURA DI TEMPERATURA ED UMIDITA' CON IL SENSORE DHT 11 con Seeeduino alimentato a
batteria; non viene inviato nulla al monitor seriale ma solo al network server
eu1.loriot.io tramite il gateway Raspberry Pi. Includere anche le librerie
-> DHT Sensor Library: https://github.com/adafruit/DHT-sensor-library
-> Adafruit Unified Sensor Lib: https://github.com/adafruit/Adafruit_Sensor
 */
//collegare il pin dati del sensore al pin digitale 2;
//NB: collegare +VCC a 3.3 V per seeeduino!!

#include <DHT.h>
#include <DHT_U.h>

#define DHTTYPE    DHT11
#define DHTPIN 2   // Collega il pin segnale al pin 2
//alimentare a 3,3 V!!!!!!

#include <LoRaWanmd.h>
//libreria LoRaWan modificata per Seeeduino alimentato a batteria

DHT dht(DHTPIN, DHTTYPE);

float temp; //temperatura
float umid; //umidità
char buffer[256];
//---------------------------------------
void setup(){
    dht.begin();  // Inizializza il sensore
    lora.init(); //inizializza il modem LoRa

    lora.setKey("lavostraNWKKEY", "lavostraAPPSKEY", NULL);
/*Inserire le proprie Chiavi NWKKEY, APPSKEY, (NULL significa che APPKEY NON viene usata
nella modalità APB, Activation By Personalization) da configurare sia su Seeeduino sia su
eu1.loriot.io per configurare una sessione ABP (Activate By Personalization) */

    lora.setDeciveMode(LWABP);
    //setta la modalità ABP (Activation By Personalization)

    lora.setDataRate(DR0, EU868); //setta il data rate DR0 (250 bit/s)
//setta i canali: con f=868.1, ecc. (sono quelli obbligatori)
    lora.setChannel(0, 868.1);
    lora.setChannel(1, 868.3);
    lora.setChannel(2, 868.5);
/*Dopo aver trasmesso in uplink un frame un end device (o Mote) deve attendere la
ricezione di una risposta; a tale scopo sono definiti due intervalli di tempo (timeslot)
detti finestre di ricezione entro cui l'end node deve attendere la ricezione di messaggi
in downlink */
    lora.setReceiceWindowFirst(0, 868.1);
    lora.setReceiceWindowSecond(869.5, DR3);

    lora.setDutyCycle(false); //il duty cycle non va configurato per la modalità ABP
    lora.setJoinDutyCycle(false);

    lora.setPower(14);//setta l'EIRP di trasmissione, il cui valore massimo è 14 dBm
}
```

[7] Nei test effettuati non ci è risultato possibile utilizzare la libreria LoRaWan.h quando Seeeduino veniva alimentato a batteria in quanto lo stesso attendeva l'apertura del monitor seriale per inviare dati e quindi richiedeva la connessione a un PC e l'apertura di Arduino IDE, per ovviare a questo problema è stata modificata la libreria.

```
void loop()
{
  delay(30*1000); // Attendi 30 s fra due misure

  lora.transferPacket("Temperatura e umidita da DHT11:");
//trasmette al network server l'indicazione sul tipo di sensore usato

  temp = dht.readTemperature(); // legge la temperatura dal sensore

  trasmette(temp);
 /*richiama la funzione che trasmette al network server, tramite il gateway Rasperry PI,
il dato di temperatura rilevato, convertito in carattere e inviato in esadecimale*/

  umid = dht.readHumidity();
//legge l'umidità; ci vogliono circa 250 ms per leggere l'umidità o la temperatura

  trasmette(umid);
/*richiama la funzione che trasmette al network server, tramite il gateway Rasperry PI,
il dato di umidità rilevato, convertito in carattere e inviato in esadecimale*/

}

//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
/*funzione che trasmette i dati al network server loriot, tramite il gateway Rasperry PI,
convertendo il tipo da float a carattere; i caratteri sono inviati in esadecimale*/

void trasmette(float datofloat){

//converte i dati acquisiti da float a carattere (a cura del prof. Giorgio Meini) --
    char buffer1[16];
    String str = String(datofloat, 2);
    str.toCharArray(buffer1,16);
    char* datoascii = buffer1;
//-----------------------------------------------------

    lora.transferPacket(datoascii);
/*trasmette il dato rilevato dal sensore al network server tramite il gateway Raspberry
Pi*/

}
```

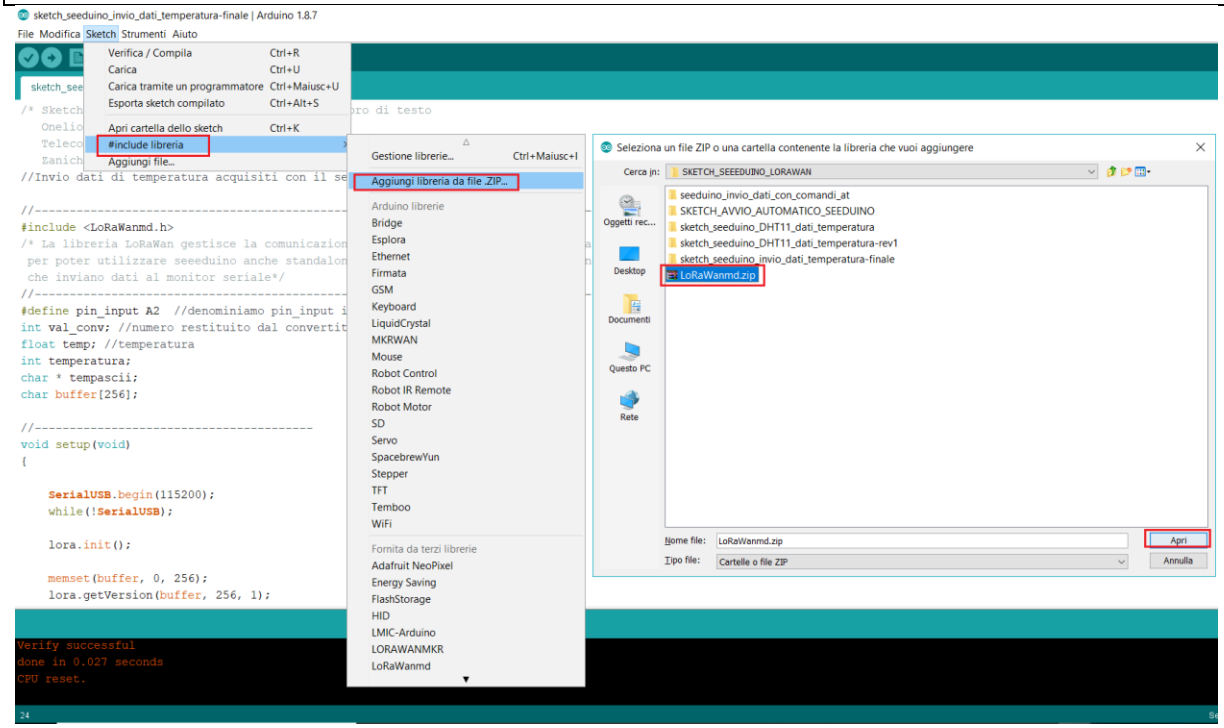


FIGURA 31 Aggiunta della libreria LoRaWanmd da file .zip.