

ELABORATO DI LAUREA IN
INGEGNERIA DELL'AUTOMAZIONE

**Piattaforma IoT per il
monitoraggio di grandezze
ambientali e inerziali basata
su sensori MEMS**

Relatore
Chiar.ma Prof.ssa
Annalisa Liccardo

Candidato
Raffaele Freschini
N39001130

Co-Relatore
Chiar.mo Prof.
Rosario Schiano Lo Moriello

Co-Relatore
Ing. Federico Gargiulo

Anno Accademico 2021/2022

*“It ain’t about how hard you hit,
it’s about how hard you can get hit
and keep moving forward.”
- Rocky Balboa*

Abstract

Il lavoro di tesi verte sul monitoraggio di grandezze fisiche ambientali ed inerziali, fondamentale per la ricerca in ambito industriale, agricolo e del climatic change, tramite l'utilizzo di dispositivi in grado di acquisire e trasmettere i dati d'interesse.

Precisamente, tramite lo sviluppo di un nodo sensore, effettuiamo le misurazioni sulle grandezze di nostro interesse e le inviamo, utilizzando la connettività adatta alla nostra applicazione (che nel nostro caso è la connettività Wi-Fi), tramite protocollo MQTT, ad un server che, ricevendo i dati formattati, può mostrarli sotto una veste grafica più pratica ed intuitiva, permettendo un'analisi immediata. Per la realizzazione di quest'ultima parte, abbiamo utilizzato una piattaforma cloud per consentire un accesso multiplatforma.

Indice

ABSTRACT.....	3
CAPITOLO 1 - IOT E MONITORAGGIO DEGLI STRUMENTI DI MISURA	5
1.1 INTRODUZIONE	5
1.2 ARCHITETTURE IoT PER SHM.....	6
1.3 PANORAMICA DELLE TECNOLOGIE IoT PER L'SHM	7
CAPITOLO 2 - BOARD B-L475E-IOT01A.....	8
2.1 INTRODUZIONE	8
2.2 MICROCONTROLLORE.....	9
2.3 SENSORI	11
LSM6DSL – 6D Giroscopio digitale e Accelerometro digitale	12
HTS221 - Sensore di Temperatura e di Umidità	13
2.4 CONNETTIVITÀ	14
Modulo Wi-Fi Inventek system ISM43362-M3G-L44	15
CAPITOLO 3 – SOFTWARE STMCUBE.....	16
3.1 CUBEMX.....	16
3.1.1 Panoramica del software	16
3.1.2 Libreria MEMS1	19
3.1.3 Modifiche effettuate	19
3.2 CUBEIDE	22
3.2.1 Panoramica dell'ide.....	22
3.2.2 Librerie.....	23
CAPITOLO 4 - DASHBOARD	24
4.1 AZURE.....	24
4.2 MQTT V3.1	25
4.2.1 Introduzione	25
4.2.2 Message format	26
4.2.3 Tipologie di Messaggi.....	27
4.3 DASHBOARD.....	29
CAPITOLO 5 – MONITORAGGIO DEI NODI SENSORE.....	30
5.1 PANORAMICA DEL CODICE UTILIZZATO	30
5.2 DASHBOARD DI AZURE.....	34
5.2.1 Creazione del device	34
5.2.2 Creazione della Dashboard.....	35
CAPITOLO 6 – SVILUPPI FUTURI	38
6.1 FFT E WAVELET TRANSFORM.....	38
6.2 AI – FORECASTING	38
6.3 DEEP LEARNING ALGORITHMS	38
BIBLIOGRAFIA	39
INDICE DELLE FIGURE	40
RINGRAZIAMENTI.....	42

Capitolo 1 - IoT e Monitoraggio degli Strumenti di Misura

1.1 Introduzione

Il monitoraggio della salute strutturale (SHM) e la valutazione dei danni delle infrastrutture di ingegneria civile sono compiti complessi. La salute strutturale e la resistenza delle strutture sono influenzate da vari fattori, come la fase di produzione del materiale, il trasporto, il posizionamento, la lavorazione e la stagionatura del calcestruzzo. I progressi tecnologici e la diffusa disponibilità di reti Wi-Fi hanno portato l'SHM a passare dai tradizionali metodi cablati a sensori wireless. È possibile effettuare una valutazione completa della salute strutturale attraverso l'uso efficiente di dati acquisiti in tempo reale, tramite sensori IoT, che monitorano diversi parametri legati alla salute strutturale e che sono accessibili attraverso sistemi di memorizzazione cloud. I dati del sensore possono essere successivamente utilizzati per varie applicazioni, come la previsione del deterioramento della costruzione in muratura, la previsione della fase iniziale di compressione e della resistenza del calcestruzzo, prevedendo il momento ottimale per la rimozione di casseforme, il rilevamento di buche sulle strade, la determinazione della qualità costruttiva e la diagnosi dei danni da corrosione. Di seguito analizziamo le applicazioni delle tecnologie IoT wireless per il monitoraggio dello stato di salute strutturale delle costruzioni ad opera dell'ingegneria civile. (Mayank Mishra, 2022)

1.2 Architetture IoT per SHM

I sistemi IoT hanno un'architettura robusta per servire gli scopi previsti dal loro utilizzo. Vengono utilizzati vari tipi di sistemi IoT nelle diverse applicazioni ingegneristiche e si basano tutte su framework e data flows simili.

Un sistema SHM basato su IoT consiste principalmente di cinque livelli:

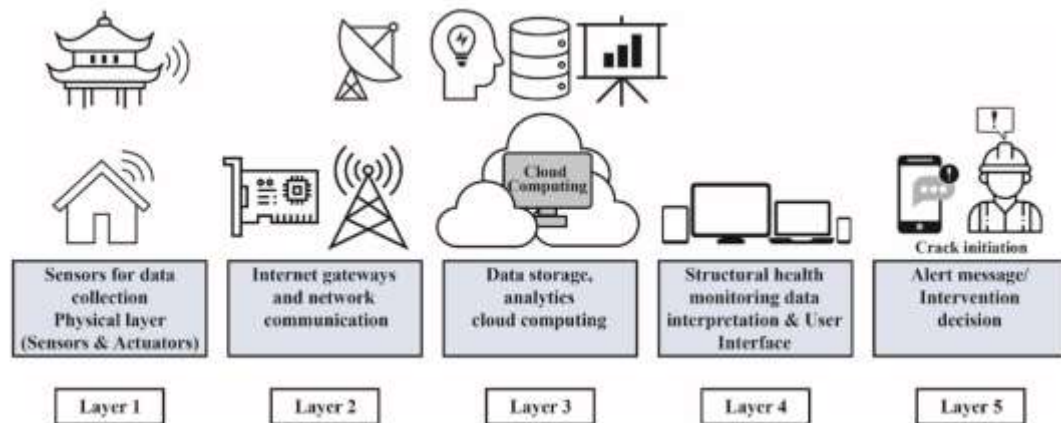


Figura 1. 1 - Architetture IoT per applicazioni SHM

Le “cose” nell'IoT possono essere, ad esempio, edifici che sono connessi ad internet, con sensori e attuatori integrati che misurano i parametri SHM e li trasmettono tramite gateway IoT. La fase successiva prevede la raccolta dei dati da parte dei sistemi di acquisizione dati, che filtrano enormi quantità di dati per effettuare ulteriori analisi. Il terzo livello analizza i dati attraverso tecniche di apprendimento automatico, per la visualizzazione di quest'ultimi. Successivamente i dati visualizzati vengono trasferiti a piattaforme basate su cloud, che analizzano tali dati per fornire approfondimenti aggiuntivi. Gli ultimi due step sono dedicati alla visualizzazione e all'informazione degli addetti ai lavori. (Mayank Mishra, 2022)

1.3 Panoramica delle tecnologie IoT per l'SHM

Questa sezione presenta una panoramica dei tipi più popolari di sensori basati su IoT utilizzati per l'SHM. Alcune proprietà monitorate, come l'indurimento e la vibrazione, potrebbero non essere direttamente correlate a proprietà meccaniche; tuttavia, questi fattori possono influenzare le prestazioni a lungo termine. (Mayank Mishra, 2022)

I dati raccolti dai sensori possono variare considerevolmente e includere, ad esempio, umidità relativa, accelerazione, video e/o audio. Possono essere utilizzati per determinare i parametri che influenzano la sicurezza e le prestazioni delle strutture. (Mayank Mishra, 2022)

Un esempio di monitoraggio molto importante è relegato al monitoraggio della resistenza alla compressione iniziale del calcestruzzo, dove vengono utilizzati sensori IoT per rilevare la temperatura all'interno di quest'ultimo. Questa temperatura è correlata con il calore dell'idratazione e l'età del calcestruzzo e può essere utilizzata con l'indice, approvato dall'istituto: "American Society for Testing and Materials", per stimare la resistenza alla compressione iniziale del calcestruzzo. Questo tipo di monitoraggio ha portato allo sviluppo e alla creazione di dispositivi impermeabili tascabili, abilitati al Wi-Fi, che possono essere facilmente incorporati in vari elementi strutturali, come travi, colonne e solai. (Mayank Mishra, 2022)

Un altro esempio di monitoraggio è il caso del rilevamento dei danni causati dalle vibrazioni, dove il danneggiamento viene valutato monitorando i cambiamenti nelle frequenze naturali e nelle forme modali. Il monitoraggio si basa proprio sulla variazione dei parametri modali sensibili. Infatti, qualsiasi modifica dei parametri modali globali, ad esempio variazioni di massa, rigidità, smorzamento, può essere identificato tramite un accelerometro. (Mayank Mishra, 2022)

Capitolo 2 - Board B-L475E-IOT01A

2.1 Introduzione

La Board utilizzata permette il monitoraggio di parametri provenienti da sensori ambientali, di vibrazioni e acustici. È adatto per il monitoraggio di impianti in cui non è possibile effettuare un collegamento cablato diretto per la ricezione dei dati.

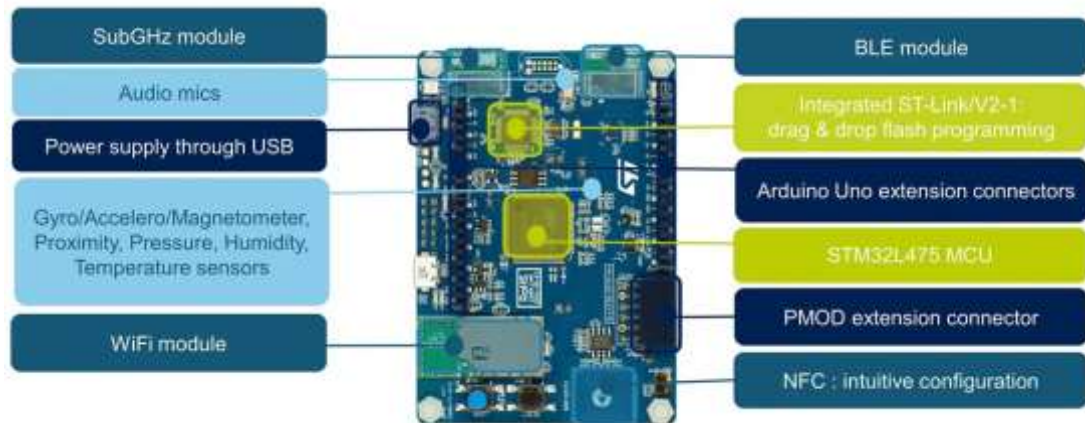


Figura 2. 1 - Schema della Board

Di seguito, vengono illustrati i componenti utilizzati per effettuare le misurazioni. (ST)

2.2 Microcontrollore

La board integra un microcontrollore STM32L475VGT6 ARM® Cortex®-M4 a 32 bit. La scheda presenta una porta MicroB USB collegata ad un connettore SWD (Serial Wire Debug) per la programmazione e il debug dell'MCU. Permette, inoltre, la trasmissione delle misurazioni effettuate.

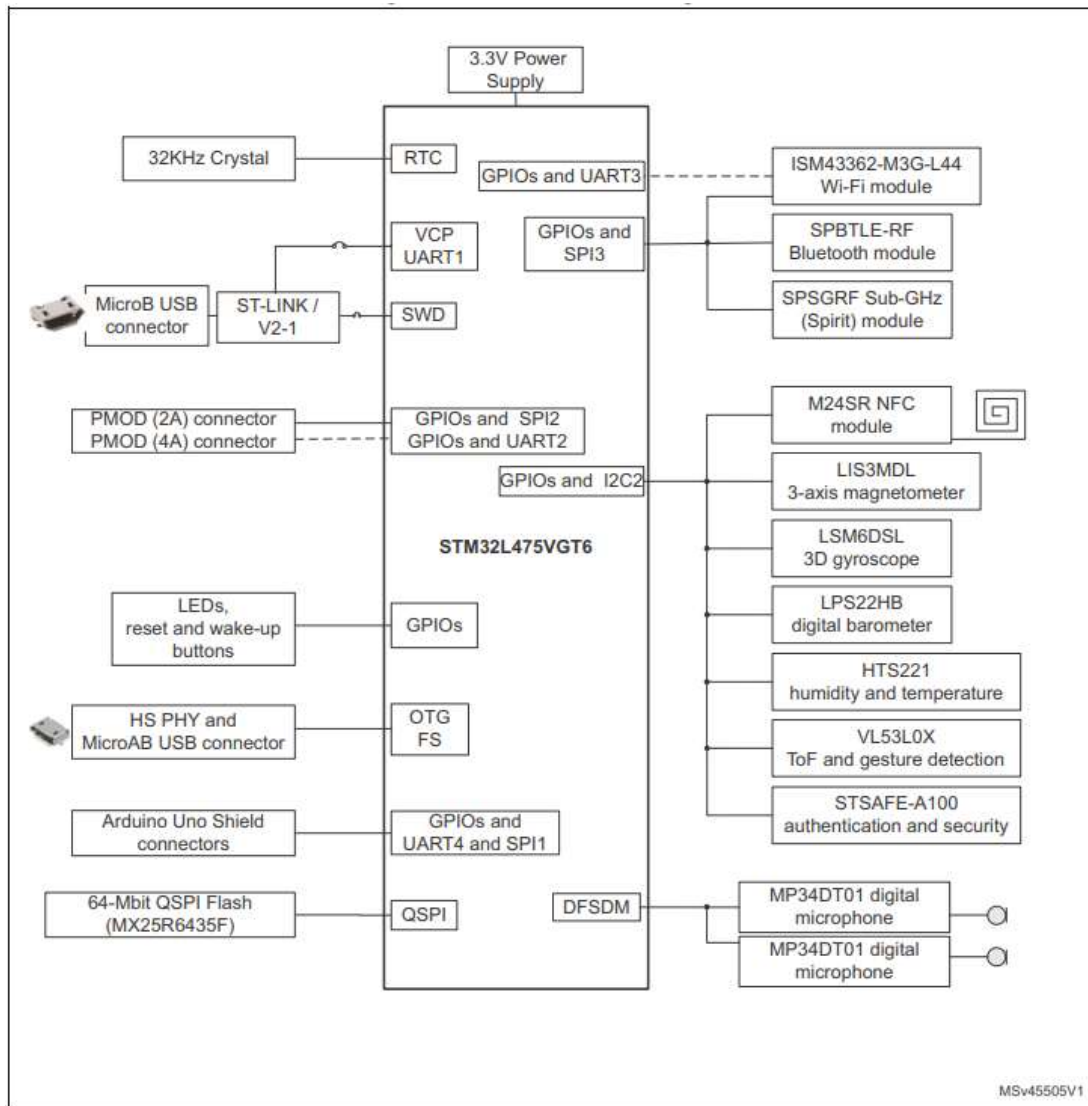


Figura 2. 2 - Hardware block diagram

Il microcontrollore opera ad una frequenza fino ad 80MHz. È dotato di un'unità a virgola mobile (FPU) a precisione singola che supporta tutte le istruzioni di elaborazione dati a singola precisione ARM e i tipi di dati.

Implementa un set completo di istruzioni DSP e una unità di protezione della memoria (MPU) che migliora la sicurezza delle applicazioni.

Ha una ricca dotazione di periferiche analogiche:

2x	ADC da 12-bit
2x	Comparatori ultra-low-power
2x	Amplificatori Operazionali
2x	DAC da 12-bit

dispone inoltre di 18 interfacce di comunicazione:

1x	USB OTG 2.0
2x	SAIs (Serial Audio Interface)
3x	I2C
6x	UART
3x	SPI
1x	Quad SPI
1x	CAN
1x	SDMMC

(ST)

2.3 Sensori

La Board incorpora diversi sensori per rilevare vibrazioni, parametri ambientali e parametri sonori. (ST)

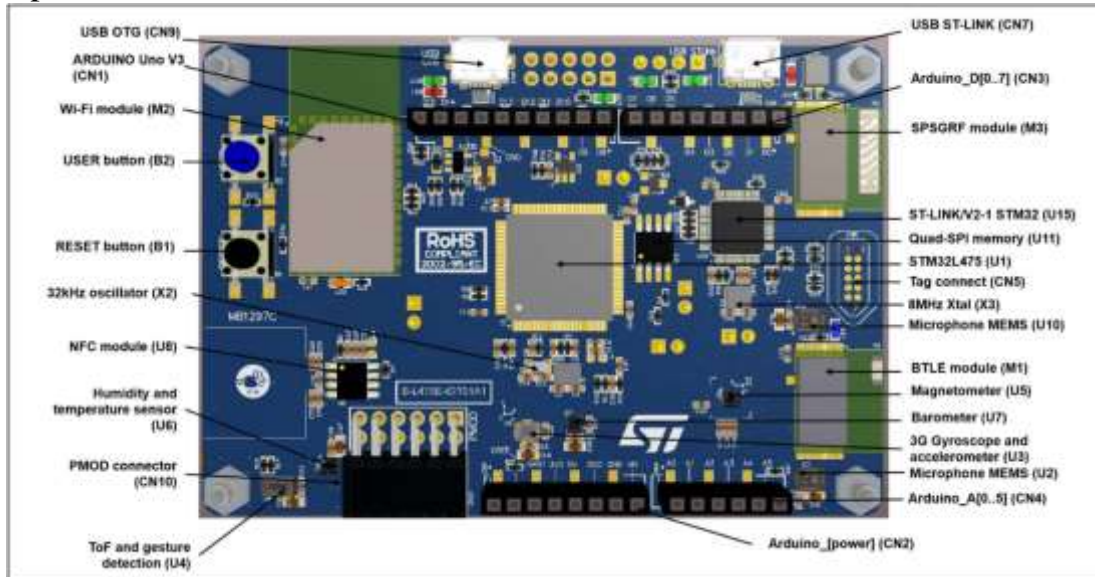


Figura 2. 3 - Top View della Board

Individuiamo i seguenti sensori:

- U3: LSM6DSL - 6G Giroscopio digitale e Accelerometro digitale
- U5: LIS3MDL - Magnetometro
- U6: HTS221 - Sensore di Temperatura e di Umidità
- U7: LPS22HB - Barometro

Analizziamo nello specifico i sensori che ci interessano principalmente.

LSM6DSL – 6D Giroscopio digitale e Accelerometro digitale

L'accelerometro LSM6DSL è un sistema composto da un Accelerometro Digitale 3D e un Giroscopio Digitale 3D che lavora a 0.65mA nella modalità ad alte prestazioni e abilitando le funzionalità always-on a basso consumo per un'ottimale esperienza lato consumatore. (ST)

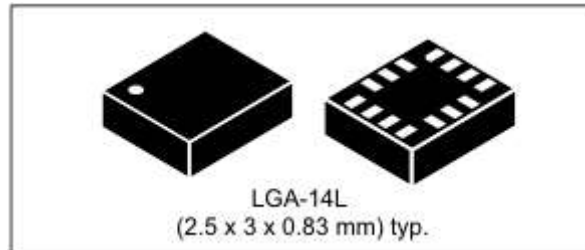


Figura 2. 4 - LSM6DSL

Supporta i principali requisiti del sistema operativo, offrendo sensori reali, virtuali e batch con 4kByte per il raggruppamento dinamico dei dati. Ha una scala completa di accelerazione $\pm 2/\pm 4/\pm 8/\pm 16$ g e un intervallo di velocità angolare di $\pm 125/\pm 250/\pm 500/\pm 1000/\pm 2000$ dps. (ST)

LA_SoDr	Linear acceleration sensitivity change vs. temperature ⁽⁴⁾	from -40° to +85°		± 0.01		%/°C
G_SoDr	Angular rate sensitivity change vs. temperature ⁽⁴⁾	from -40° to +85°		± 0.007		%/°C

Figura 2. 5 - Sensitivity Acc + Gyro

L'elevata robustezza agli urti meccanici lo rende la scelta preferita dei progettisti di sistemi per la creazione e la fabbricazione di prodotti affidabili.

Part number	Temp. range [°C]	Package	Packing
LSM6DSL	-40 to +85	LGA-14L (2.5x3x0.83mm)	Tray
LSM6DSLTR	-40 to +85		Tape & Reel

Figura 2. 6 - Device Summary

Applicazioni:

- Motion tracking e gesture detection
- Collecting sensor data
- Indoor navigation
- IoT e connected devices
- Intelligent power saving per dispositivi palmari
- Monitoraggio delle vibrazioni e compensazione

HTS221 - Sensore di Temperatura e di Umidità

HTS221 è un sensore ultracompatto per umidità relativa e temperatura. Include un elemento di rilevamento e un ASIC a segnale misto per fornire le informazioni di misura attraverso interfacce seriali digitali. (ST)

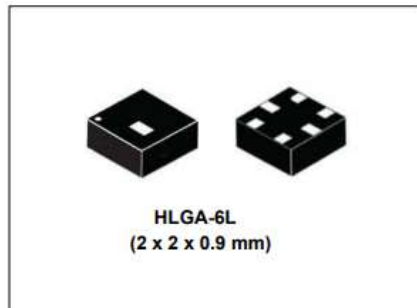


Figura 2. 7 - HTS221

L'elemento sensibile è costituito da una struttura a condensatore planare dielettrico polimerico in grado di rilevare le variazioni di umidità relativa ed è prodotto utilizzando un processo ST dedicato. (ST)

Symbol	Parameter	Test condition	Min.	Typ. ⁽¹⁾	Max.	Unit
H _{op}	Operating humidity range		0	–	100	% rH
H _{bit}	Humidity output data			16	–	bit
H _s	Humidity sensitivity			0.004		%rH/LSB
				256		LSB/%rH
H _{acc}	Humidity accuracy ⁽²⁾	20 to 80% rH		±3.5		% rH
		0 to 100% rH		±5		
H _{noise}	Humidity noise ⁽³⁾			0.03		RMS
H _{hys}	Humidity hysteresis			±1		% rH
H _{step}	Humidity response time ⁽⁴⁾	t @ 63%		10		s
H _{drift}	Humidity long-term drift	20 to 80% rH		0.5		%rH/yr
T _{op}	Operating temperature range		-40	–	120	°C
T _{bit}	Temperature output data		–	16	–	bit
T _s	Temperature sensitivity			0.016		°C/LSB
				64		LSB/°C
T _{acc}	Temperature accuracy	15 to 40 °C		±0.5		°C
		0 to 60 °C		±1		
T _{noise}	Temperature noise ⁽³⁾			0.007		RMS
T _{step}	Temperature response time	t @ 63%		15		s
T _{drift}	Temperature long-term drift	T = 0 to 80 °C			0.05	°C/yr
ODR	Humidity and temperature digital output data rate			1/7/12, 5		Hz

Figura 2. 8 - Sensitivity HTS221

2.4 Connettività

La board presenta diverse possibilità di connessione con cui interfacciarsi:

- USB OTG 2.0
- NFC
- BTE (Bluetooth Low-Energy)
- Wi-Fi
- SubGHz

La differenza più tangibile tra queste diverse tecnologie è il range di azione, dove primeggia la SubGHz con un range che va anche oltre 1Km.

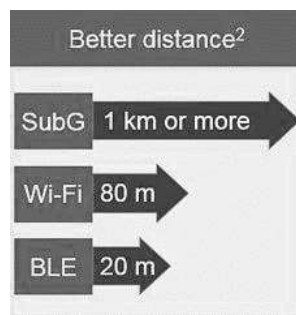


Figura 2. 9 - Better distance

Nel nostro caso, per una maggior velocità e praticità di testing abbiamo usato la connettività di tipo Wi-Fi, nonostante sia peggiore in termini di consumo e di range extension, rispetto alla SubGHz. Grazie al protocollo MQTT, si può facilmente cambiare tipologia di connettività, passando alla SubGHz, grazie alla quale è possibile inviare le misure rilevate per il monitoraggio della salute delle strutture a distanze più elevate.

Modulo Wi-Fi Inventek system ISM43362-M3G-L44

Il modulo ISM43362-M3G-L44 (M2) del sistema Inventek è implementato sul lato superiore della board. (Inventek System)



Figura 2. 10 - ISM43362-M3G-L44

Questo è un modulo wireless integrato (eS-WiFi) e l'hardware è costituito da un Arm® Cortex® -M3 con processore host STM32, un'antenna integrata (o un'antenna esterna opzionale) e con un dispositivo Wi-Fi Broadcom. Il modulo utilizza una connessione di tipo UART (UART3 nel caso del STM32L475VG) o un'interfaccia SPI (SPI3 nel caso di STM32L475VG). È selezionata l'interfaccia SPI per un output data rate maggiore, con il firmware corrispondente già precaricato. Il modulo Wi-Fi non richiede, dunque, alcun sistema operativo ed ha uno Stack TCP/IP completamente integrato che richiede solo comandi AT per stabilire la connettività wireless. (Inventek System)

Capitolo 3 – Software STMCube

In questo capitolo approfondiamo i software utilizzati. Tutti i software utilizzati fanno parte della suite di ST, scaricabile direttamente dal sito ufficiale.

3.1 CubeMX

3.1.1 Panoramica del software

STM32CubeMX è uno strumento grafico che consente una configurazione di microcontrollori e microprocessori STM32, nonché permette la generazione del codice C di inizializzazione corrispondente per processori della famiglia Arm® Cortex®-M core o un parziale Linux® Device Tree per processori della famiglia Arm® Cortex®-A core, attraverso un processo graduale.

Il primo passo consiste nella selezione di un microcontrollore STM32 di STMicroelectronics, di un microprocessore o di una piattaforma di sviluppo che corrisponda al set di periferiche richiesto o di un esempio eseguito su una specifica piattaforma di sviluppo.

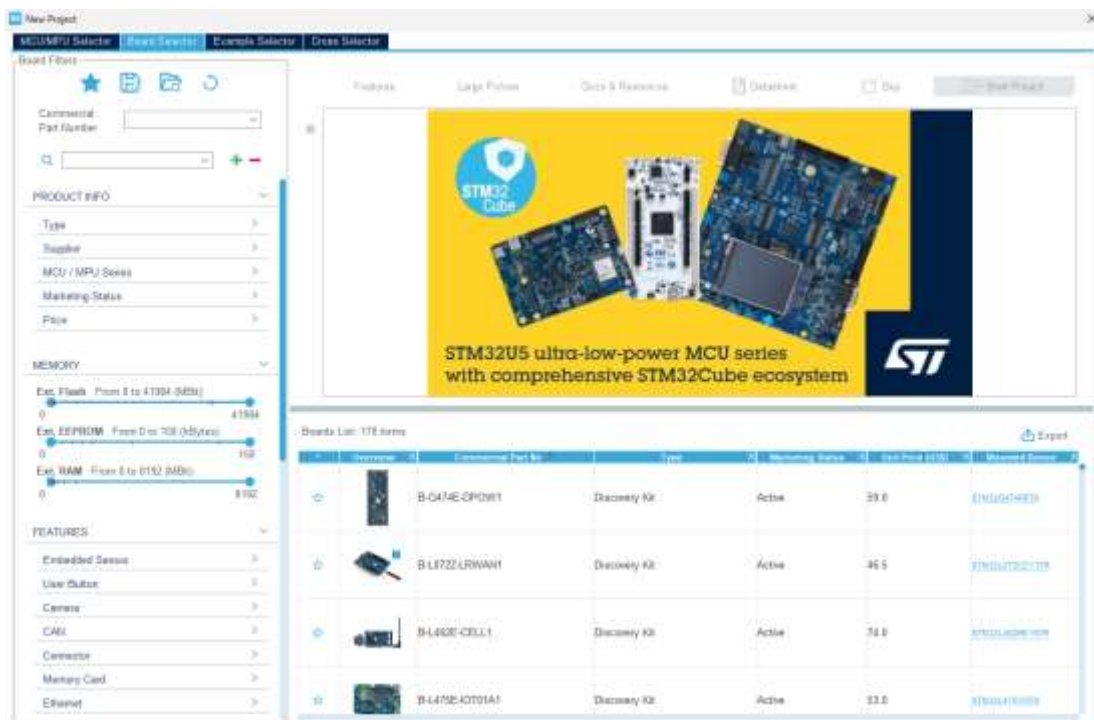


Figura 3. 1 - Board Selection

The screenshot displays the STM32CubeMX software interface. The central window shows a top-down view of the STM32L475VGTx microcontroller chip. The chip is labeled with the ST logo, the part number 'STM32L475VGTx', and the package type 'LQFP100'. The pins are color-coded: green for I/O pins, yellow for power pins, and blue for other pins. The left sidebar contains a 'Pinout & Configuration' panel with a search bar and a list of pins. The top bar shows 'Pinout & Configuration' and 'Project Manager' tabs.

Figura 3. 3 - Pinout & Configuration

Package / Family / Component	Status	Version	Selection
Rowlett's X-CUBE-USBDMIT00		5.5.8.4 (3.4)	9040
SEGGER X-CUBE-usbD0		5.5.1.0	9040
STMicroelectronics FP-AT8860T0A1	2	5.5.1.0	9040
STMicroelectronics FP-ATR88F081		5.5.8.0	9040
STMicroelectronics X-CUBE-AI		7.5.0.0	9040
STMicroelectronics X-CUBE-AI00B0LD		1.3.0.0	9040
STMicroelectronics X-CUBE-AI8		5.8.1.0	9040
STMicroelectronics X-CUBE-AI77000-P4	2	5.5.1.0	9040
STMicroelectronics X-CUBE-AI77000-P2	2	5.5.0.0	9040
STMicroelectronics X-CUBE-AI77000-S0	3	5.5.0.0	9040
STMicroelectronics X-CUBE-AI77000-GA	3	5.5.0.0	9040
STMicroelectronics X-CUBE-AI77000-A0	3	5.5.0.0	9040
STMicroelectronics X-CUBE-AI77000-LA		5.8.0	
STMicroelectronics X-CUBE-AI77000-G3	2	5.5.0.0	9040
STMicroelectronics X-CUBE-AI77000-P0	3	5.5.0.0	9040
STMicroelectronics X-CUBE-AI77000-H0	3	5.5.0.0	9040
STMicroelectronics X-CUBE-BLE1		6.2.3.0	9040
STMicroelectronics X-CUBE-BLE2		3.3.0.0	9040
STMicroelectronics X-CUBE-BLE0R		1.3.0.0	9040
STMicroelectronics X-CUBE-DISPLAY		2.0.0.0	9040
STMicroelectronics X-CUBE-DS18B01		3.1.1.0	9040
STMicroelectronics X-CUBE-DS001		6.0.0.0	9040
STMicroelectronics X-CUBE-GPU		5.5.0.0	9040
STMicroelectronics X-CUBE-MEM01		5.4.0.0	9040
STMicroelectronics X-CUBE-MFC3		2.0.0.0	9040
STMicroelectronics X-CUBE-MPC7		1.0.0.0	9040
STMicroelectronics X-CUBE-SPI000-P1		4.0.0.0	9040
STMicroelectronics X-CUBE-SUNCE		5.0.0.0	9040
STMicroelectronics X-CUBE-TOP1		5.5.0.0	9040

Figura 3. 4 - Software Packs Component Selector

Inoltre, una utility unica nella suite STM32CubeMX, STM32PackCreator, aiuta gli sviluppatori a creare i propri pacchetti di espansione STM32Cube avanzati.

Alla fine l'utente avvia la generazione del codice di inizializzazione che corrisponde alle scelte di configurazione selezionate, pronto per essere utilizzato all'interno di diversi ambienti di sviluppo, o a partial Linux® Device Tree per i processori della famiglia Arm® Cortex®-A.

STM32CubeMX viene fornito all'interno di STM32Cube. (ST)

3.1.2 Libreria MEMS1

Nel nostro caso abbiamo aggiunto come software package il pacchetto dedicato ai sensori MEMs. Il seguente pacchetto viene eseguito su STM32 e include driver che riconoscono i sensori e raccolgono dati di temperatura, umidità, pressione e movimento. L'espansione si basa sulla tecnologia software STM32Cube per facilitare la portabilità tra diversi microcontrollori STM32. Il software viene fornito con un'implementazione di esempio dei driver in esecuzione sulle schede di espansione X-NUCLEO-IKS01A2/X-NUCLEO-IKS01A3/X-NUCLEO-IKS02A1 collegate a una scheda di sviluppo STM32 Nucleo. Il software è disponibile anche su GitHub, dove gli utenti possono segnalare bug e proporre nuove idee attraverso le schede Problemi e Richieste di pull.


▼ STMMicroelectronics.X-CUBE-MEMS1	✓ 	9.3.0 ▼		
> Device MEMS1_Applications		9.3.0		
▼ Board Part AccGyr	✓	5.4.0		
LSM6DSL	✓	5.4.0	I2C ▼	
LSM6DSO			Not selected ▼	

Figura 3. 5 - Software Package MEMS1

Precisamente abbiamo aggiunto solo le parti dedicate all'accelerometro LSM6DSL e al sensore di umidità e temperatura HTS221, scegliendo accuratamente la connessione I2C.

3.1.3 Modifiche effettuate

Di seguito presento le modifiche che abbiamo effettuato lato pinout. Abbiamo innanzitutto attivato la connessione I2C per utilizzare l'accelerometro e il sensore di umidità e temperatura.

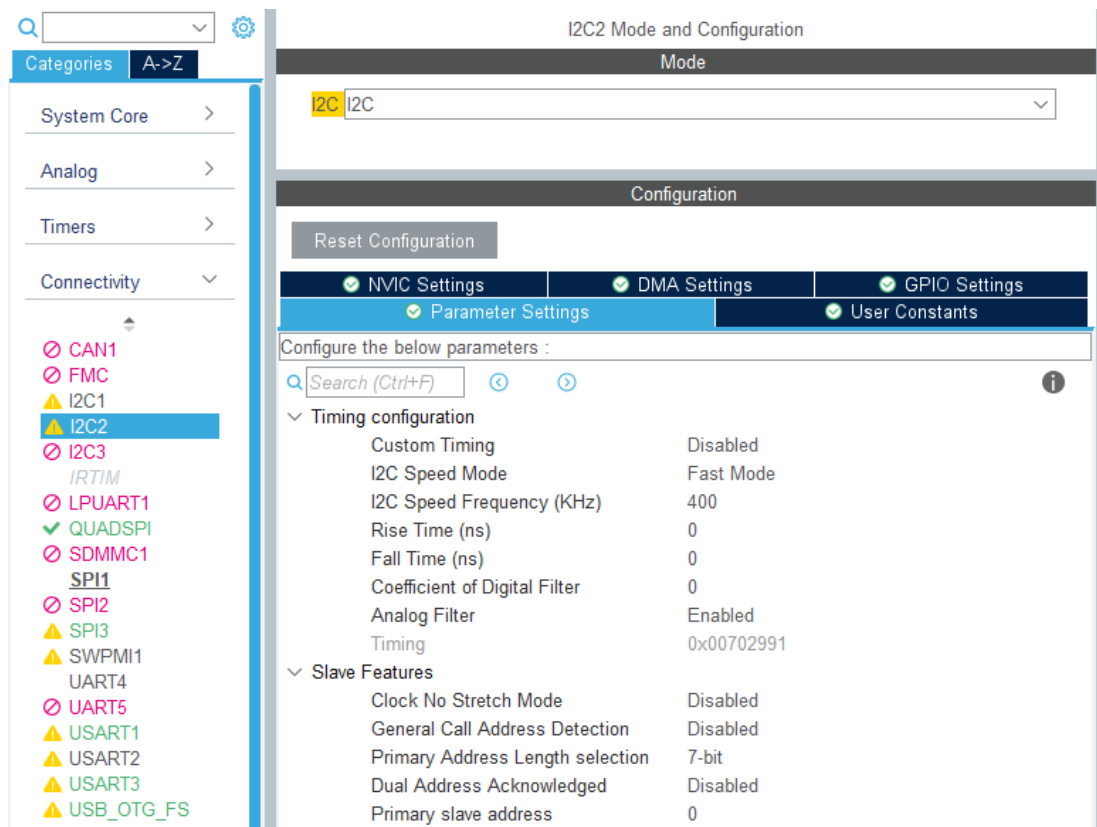


Figura 3. 6 - I2C Configuration

Successivamente, abbiamo attivato la connessione SPI3, per il modulo Wi-Fi, e

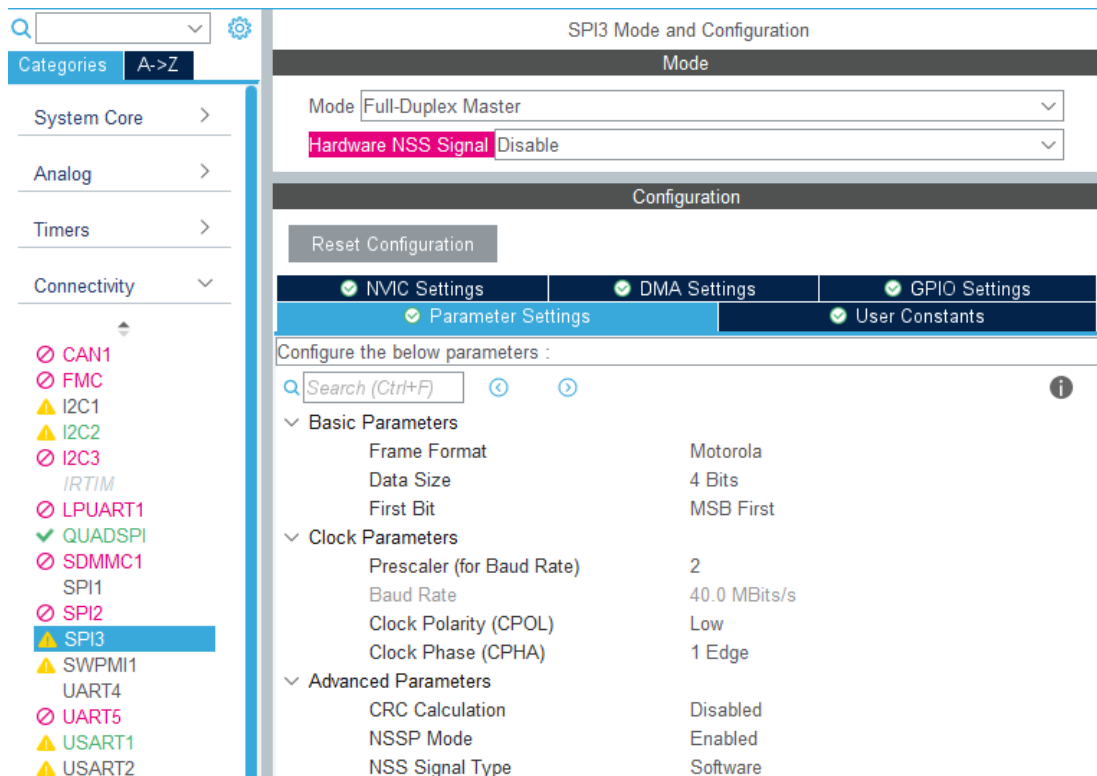


Figura 3. 7 - SPI3 Configuration

la connessione USART1 per la connessione via USB, in modo da leggere i dati da terminale.

Lato ‘Clock Configuration’ abbiamo impostato il clock a 80MHz, come si può vedere nella Figura 3.2, e attivato e configurato il Timer 2 (TIM2), nella sezione ‘Pinout’, in modo da avere un interrupt ogni secondo.

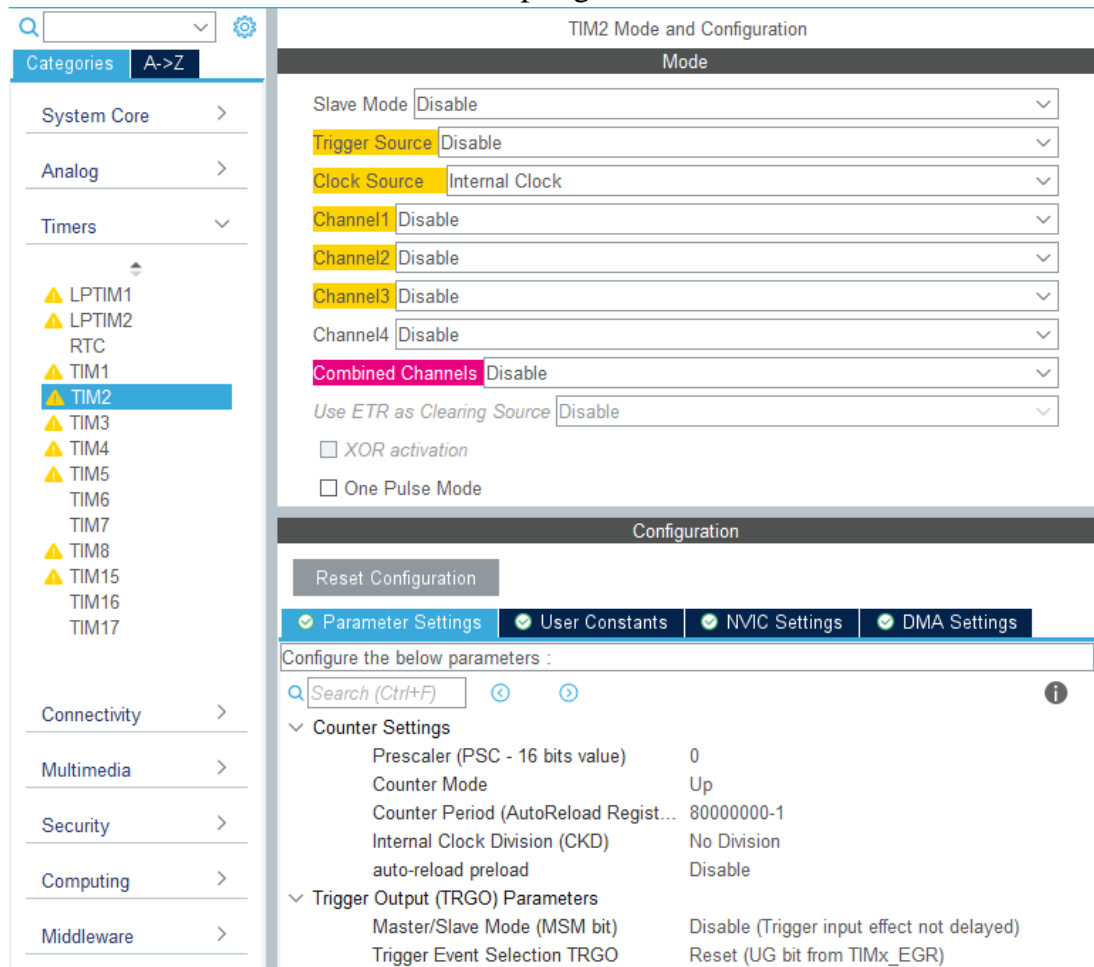


Figura 3. 8 - Tim2 Configuration

3.2 CubeIde

3.2.1 Panoramica dell'ide

STM32CubeIDE è uno strumento di sviluppo multi-OS all-in-one, che fa parte dell'ecosistema software STM32Cube.

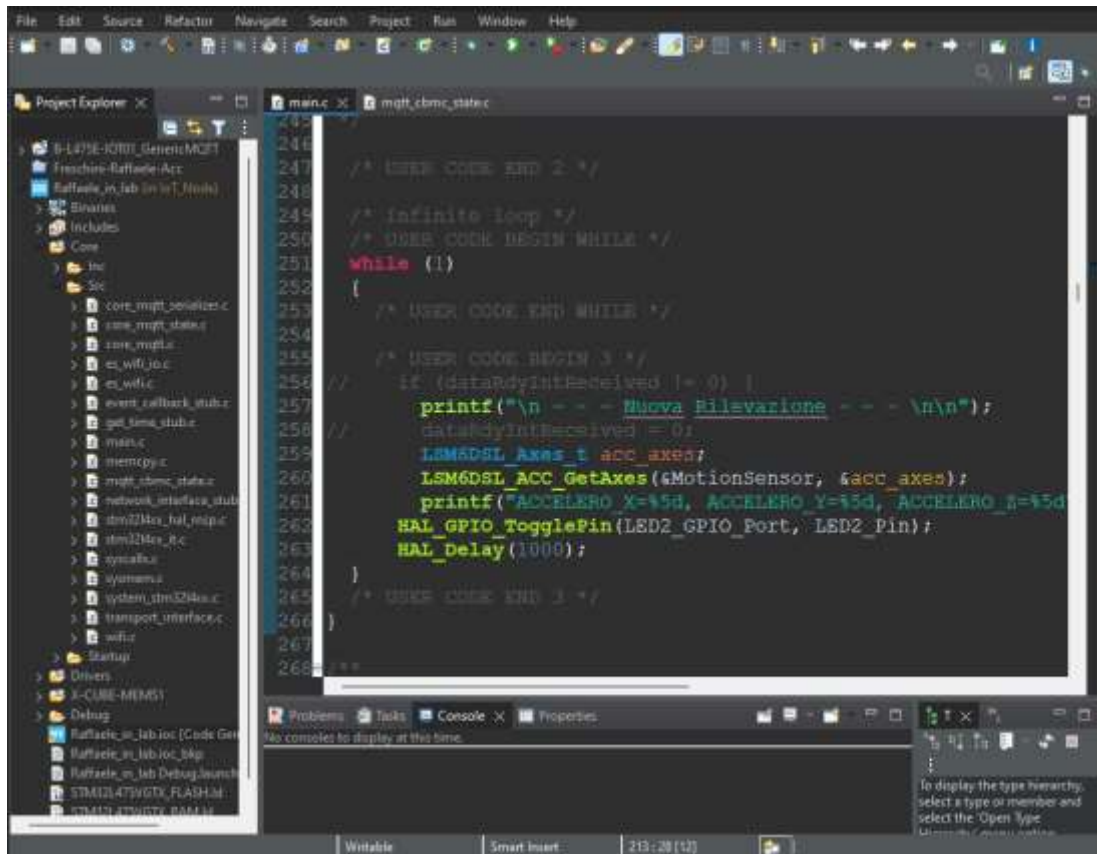


Figura 3. 9 - Panoramica Ide

STM32CubeIDE è una piattaforma di sviluppo C/C++ avanzata avente la configurazione delle periferiche, la generazione di codice, la compilazione di codice e funzionalità di debug per microcontrollori e microprocessori STM32. Si basa sul framework Eclipse®/CDT™ e sulla toolchain GCC per lo sviluppo e GDB per il debug. Consente l'integrazione di centinaia di plugin esistenti che completano le funzionalità dell'IDE Eclipse®.

STM32CubeIDE integra le funzionalità di configurazione e creazione di progetti STM32 da STM32CubeMX per offrire un'esperienza di strumento all-in-one e risparmiare tempo di installazione e sviluppo. Dopo la selezione di un MCU o MPU STM32 vuoto, o di un microcontrollore o microprocessore preconfigurato dalla selezione di una scheda o dalla selezione di un esempio, viene creato il progetto e generato il codice di inizializzazione. (ST)

In qualsiasi momento durante lo sviluppo, l'utente può tornare all'inizializzazione e alla configurazione delle periferiche o del middleware e rigenerare il codice di inizializzazione senza alcun impatto sul codice utente.

STM32CubeIDE include analizzatori di build e stack che forniscono all'utente informazioni utili sullo stato del progetto e sui requisiti di memoria. Inoltre include anche funzionalità di debugging standard e avanzate, tra cui visualizzazioni di registri core della CPU, memorie e registri periferici, oltre a monitoraggio variabile in tempo reale, interfaccia Serial Wire Viewer o analizzatore di guasti. (ST)

3.2.2 Librerie

Per lavorare al meglio abbiamo integrato la libreria Wi-Fi e la libreria MQTT v3.1 di [ST](#) per poter gestire in modo robusto la comunicazione con il server.

Capitolo 4 - Dashboard

4.1 Azure

Microsoft Azure è la piattaforma cloud pubblica di Microsoft, che offre servizi di cloud computing. Tramite Azure vengono erogati servizi appartenenti a diverse categorie quali: risorse di elaborazione, archiviazione, memorizzazione, trasmissione dati e interconnessioni di reti, analisi, intelligence, apprendimento automatico, sicurezza e gestione delle identità, monitoraggio e gestione, nonché servizi per lo sviluppo di applicazioni.

Il numero e il tipo di servizi erogati vengono modificati da Microsoft con cadenza periodica.

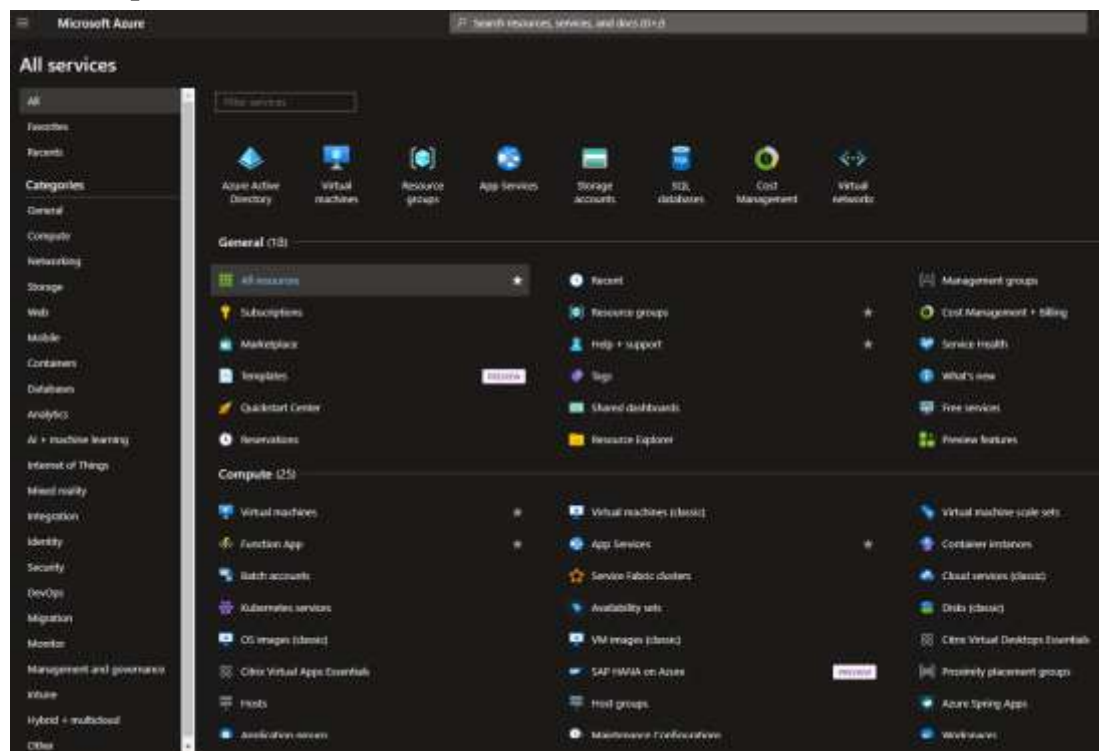


Figura 4. 1 - Schermata All Services di Microsoft Azure

I servizi messi a disposizione da Microsoft Azure possono essere classificati in tre aree, a seconda della modalità di erogazione adottata:

- Infrastructure as a Service (IaaS)
- Platform as a Service (PaaS)
- Software as a Service (SaaS)

Fornisce inoltre servizi di mobile Backend as a Service (mBaaS).

Ciascun servizio prevede un pagamento in base al consumo e le modalità con cui ne viene determinato il costo sono specifiche per il servizio stesso.

La rete di datacenter da cui sono erogati i servizi è costituita da un numero non precisato di centri di elaborazione, raggruppati in 34 aree geografiche.

4.2 MQTT V3.1

4.2.1 Introduzione

MQ Telemetry Transport (MQTT) è un protocollo di messaggistica leggero basato su broker Publish/Subscribe, progettato per essere aperto, semplice, leggero e facile da implementare (MQTT).

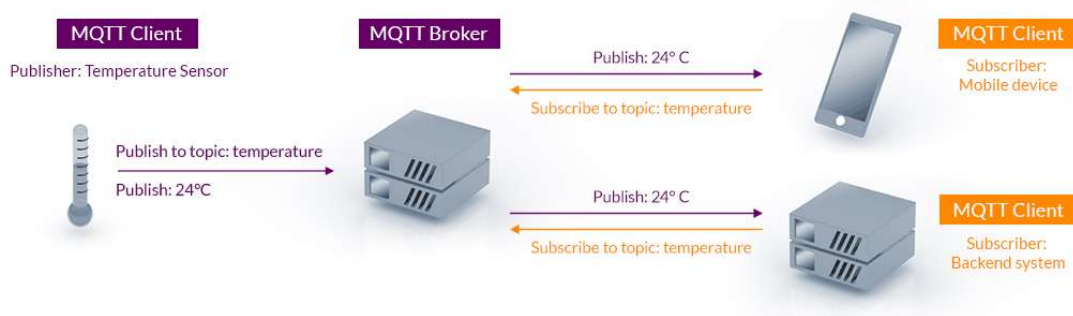


Figura 4. 2 - Schema MQTT

Queste caratteristiche lo rendono ideale per l'utilizzo in ambienti ristretti, ad esempio, ma non solo:

- dove la rete è costosa, ha una larghezza di banda ridotta o è inaffidabile;
- quando viene eseguito su un dispositivo integrato con processore o risorse di memoria limitate.

Le caratteristiche del protocollo includono:

- Il modello di pubblicazione/sottoscrizione per fornire la distribuzione dei messaggi uno a molti e il disaccoppiamento delle applicazioni;
- Un trasporto di messaggistica indipendente dal contenuto del payload;
- L'uso di TCP/IP per fornire connettività di rete di base;
- Tre qualità di servizio per la consegna dei messaggi:
 - "Al massimo una volta", in cui i messaggi vengono consegnati secondo i migliori sforzi della rete TCP/IP sottostante. Può verificarsi la perdita o la duplicazione dei messaggi. Questo livello potrebbe essere utilizzato, ad esempio, con i dati dei sensori ambientali in cui non importa se una singola lettura viene persa poiché quella successiva verrà pubblicata subito dopo.

- "Almeno una volta", dove è garantito l'arrivo dei messaggi ma possono verificarsi duplicati.
- "Esattamente una volta", dove è garantito che il messaggio arrivi esattamente una volta. Questo livello potrebbe essere utilizzato, ad esempio, con sistemi di fatturazione in cui i messaggi duplicati o persi potrebbero comportare l'applicazione di addebiti errati.
- Un piccolo sovraccarico di trasporto (l'intestazione a lunghezza fissa è di soli 2 byte) e gli scambi di protocollo ridotti al minimo per ridurre il traffico di rete.
- Un meccanismo per notificare alle parti interessate una disconnessione anomala di un cliente utilizzando la funzione Last Will and Testament.

4.2.2 Message format

L'intestazione del messaggio, per ogni messaggio di comando MQTT, contiene un'intestazione fissa. Alcuni messaggi richiedono anche un'intestazione variabile e un payload.

L'intestazione fissa segue il seguente schema:

bit	7	6	5	4	3	2	1	0
byte 1	Message Type				DUP flag	QoS level		RETAIN
byte 2	Remaining Length							

Figura 4. 3 - Fixed Header

Il byte 1 contiene i campi relativo al tipo del messaggio e vari flag (DUP, livello QoS e RETAIN). Il byte 2, che è almeno un byte, contiene il campo che indica la lunghezza del pacchetto restante.

I campi sono descritti nelle sezioni seguenti. Tutti i valori dei dati sono in ordine big-endian. Una parola a 16 bit viene rappresentata come Most Significant Byte (MSB), seguita da Least Significant Byte (LSB).

I seguenti tipi di messaggi di comando MQTT hanno un payload:

- COLLEGARE: il payload contiene una o più stringhe con codifica UTF-8. Specificano un identificatore unico per il client, un argomento Will, un messaggio e il nome utente e la password da utilizzare. Tutti tranne il primo sono facoltativi e la loro presenza è determinata in base ai flag nell'intestazione della variabile.
- SOTTOSCRIVI: il payload contiene un elenco di nomi di argomenti a cui il client può iscriversi e il livello di QoS. Queste stringhe sono codificate in UTF.

- SUBBACK: il payload contiene un elenco di livelli QoS concessi. Questi sono i livelli di QoS ai quali gli amministratori del server hanno consentito al client di sottoscrivere un particolare Topic Name. I livelli di QoS concessi sono elencati nello stesso ordine dei nomi degli argomenti nel messaggio SUBSCRIBE corrispondente.

La parte del payload di un messaggio PUBLISH contiene solo dati specifici dell'applicazione. Non vengono fatte ipotesi sulla natura o sul contenuto dei dati e questa parte del messaggio viene trattata come un BLOB.

Se si desidera che un'applicazione applichi la compressione ai dati del payload, è necessario definire nell'applicazione i campi flag appropriati del payload per gestirne i dettagli. Non è possibile definire flag specifici dell'applicazione nelle intestazioni fisse o variabili.

L'identificativo del messaggio è presente nell'intestazione della variabile dei seguenti messaggi MQTT: PUBLISH, PUBACK, PUBREC, PUBREL, PUBCOMP, SUBSCRIBE, SUBACK, UNSUBSCRIBE, UNSUBACK.

Il campo Message Identifier (Message ID) è presente solo nei messaggi in cui i bit QoS, nell'intestazione fissa, indicano i livelli QoS 1 o 2. L'ID del messaggio è un numero intero senza segno a 16 bit che deve essere univoco tra l'insieme di messaggi "in volo" in una particolare direzione di comunicazione. In genere aumenta esattamente di uno da un messaggio all'altro, ma non è necessario farlo.

Un client manterrà il proprio elenco di ID messaggio separato dagli ID messaggio utilizzati dal server a cui è connesso. È possibile che un client invii una PUBLISH con Message ID 1 contemporaneamente alla ricezione di una PUBLISH con Message ID 1. Non bisogna utilizzare l'ID messaggio 0 che è riservato per indicare il caso di ID messaggio non valido. (MQTT)

4.2.3 Tipologie di Messaggi

Di seguito vengono illustrati le varie tipologie di messaggio:

- Connect - quando viene stabilita una connessione socket TCP/IP da un client a un server, è necessario creare una sessione a livello di protocollo utilizzando un flusso CONNECT.
- Connack - il messaggio CONNACK è il messaggio inviato dal server in risposta a una richiesta CONNECT da parte di un client.
- Publish - un messaggio PUBLISH viene inviato da un client a un server per la distribuzione agli abbonati interessati. Ogni messaggio PUBLISH

è associato a un nome di un argomento (noto anche come Oggetto o Canale). Questo è uno spazio dei nomi gerarchico che definisce una tassonomia di fonti di informazioni per le quali gli abbonati possono registrare un interesse. Un messaggio pubblicato con un nome di un argomento specifico viene recapitato ai sottoscrittori connessi per quell'argomento. Se un client si iscrive a uno o più argomenti, qualsiasi messaggio pubblicato su tali argomenti viene inviato dal server al client come messaggio PUBLISH.

- PubAck - un messaggio PUBACK è la risposta a un messaggio PUBLISH con livello QoS 1. Un messaggio PUBACK viene inviato da un server in risposta a un messaggio PUBLISH da un client di pubblicazione e da un abbonato in risposta a un messaggio PUBLISH dal server.
- PubRec - un messaggio PUBREC è la risposta a un messaggio PUBLISH con livello QoS 2. È il secondo messaggio del flusso di protocollo QoS livello 2. Un messaggio PUBREC viene inviato dal server in risposta a un messaggio PUBLISH da un client di pubblicazione o da un sottoscrittore in risposta a un messaggio PUBLISH dal server.
- Subscribe - il messaggio SUBSCRIBE consente a un client di registrare un interesse per uno o più nomi di argomento con il server. I messaggi pubblicati in questi argomenti vengono consegnati dal server al client come messaggi PUBLISH. Il messaggio SUBSCRIBE specifica anche il livello di QoS al quale l'abbonato desidera ricevere i messaggi pubblicati.
- SubAck - un messaggio SUBACK viene inviato dal server al client per confermare la ricezione di un messaggio SUBSCRIBE. Un messaggio SUBACK contiene un elenco di livelli QoS concessi. L'ordine dei livelli di QoS concessi nel messaggio SUBACK corrisponde all'ordine dei nomi degli argomenti nel corrispondente messaggio SUBSCRIBE.
- Unsubscribe - un messaggio UNSUBSCRIBE viene inviato dal client al server per annullare l'iscrizione agli argomenti denominati.
- UnsubAck - il messaggio UNSUBACK viene inviato dal server al client per confermare la ricezione di un messaggio UNSUBSCRIBE.
- Disconnect - il messaggio DISCONNECT viene inviato dal client al server per indicare che sta per chiudere la sua connessione TCP/IP. Ciò consente una disconnessione pulita, piuttosto che abbandonare semplicemente la linea. Se il client si è connesso con il flag di sessione pulita impostato, tutte le informazioni precedentemente gestite sul client verranno eliminate. Un server non dovrebbe fare affidamento sul client

per chiudere la connessione TCP/IP dopo aver ricevuto un DISCONNECT.

4.3 Dashboard

Una dashboard è un report interattivo che si aggiorna in tempo reale e che permette di visualizzare dati provenienti da diverse fonti.

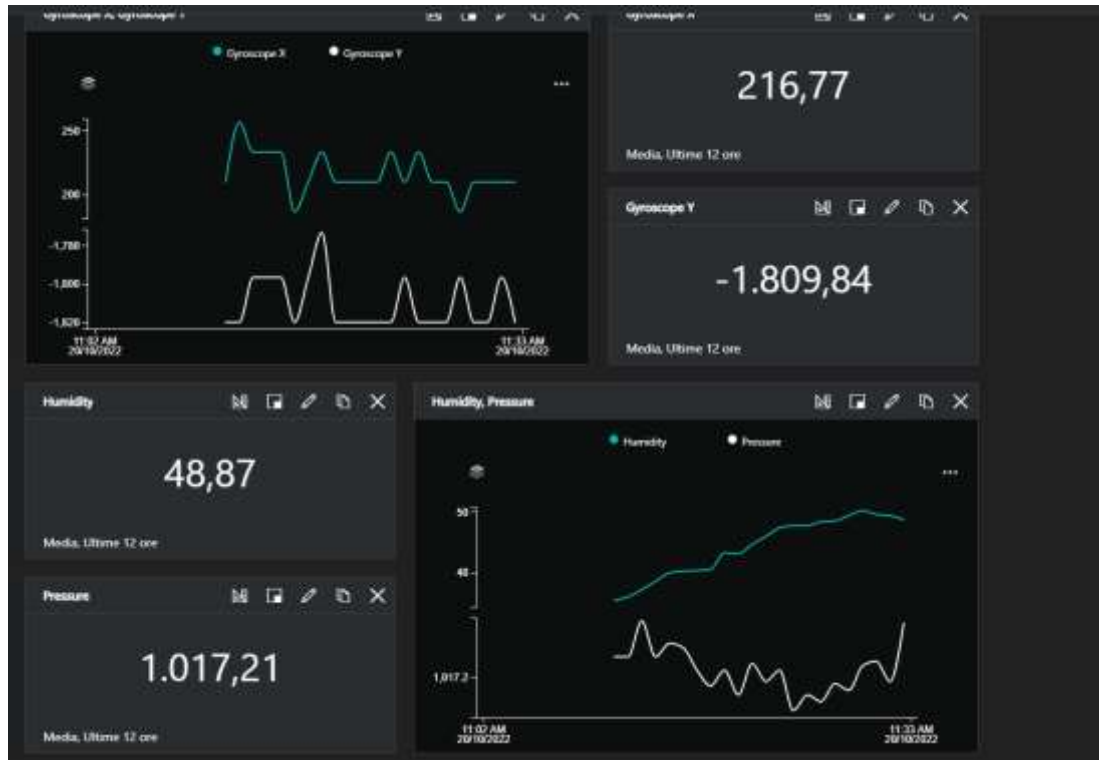


Figura 4. 4 - Dashboard Example

Rappresentano uno dei principali output del processo di digital analytics. Infatti, una volta configurato correttamente il sistema di tracking, una delle azioni più incisive che si può fare è la realizzazione di una dashboard per visualizzare i dati di più nodi sensore, contemporaneamente.

Il dettaglio implementativo della dashboard è possibile trovarlo nel capitolo successivo.

Capitolo 5 – Monitoraggio dei Nodi Sensore

5.1 Panoramica del codice utilizzato

Una volta configurato le periferiche necessarie al nostro lavoro, come specificato nel terzo capitolo, procediamo con l'inizializzazione delle periferiche.

```
/* Initialize all configured peripherals */
MX_GPIO_Init();
MX_DFSDM1_Init();
MX_QUADSPI_Init();
MX_SPI3_Init();
MX_USART1_UART_Init();
MX_USART3_UART_Init();
MX_USB_OTG_FS_PCD_Init();
MX_TIM2_Init();
/* USER CODE BEGIN 2 */
MEMS_Init();
```

Figura 5. 1 - Inizializzazione

Dove la funzione MEMS_Init() è specificata come segue:

```
/* USER CODE BEGIN 4 */
static void MEMS_Init(void)
{
    LSM6DSL_IO_t io_ctx;
    uint8_t id;
    LSM6DSL_AxesRaw_t axes;

    /* Link I2C functions to the LSM6DSL driver */
    io_ctx.BusType = LSM6DSL_I2C_BUS;
    io_ctx.Address = LSM6DSL_I2C_ADD_L;
    io_ctx.Init = BSP_I2C2_Init;
    io_ctx.DeInit = BSP_I2C2_DeInit;
    io_ctx.ReadReg = BSP_I2C2_ReadReg;
    io_ctx.WriteReg = BSP_I2C2_WriteReg;
    io_ctx.GetTick = BSP_GetTick;
    LSM6DSL_RegisterBusIO(&MotionSensor, &io_ctx);

    /* Read the LSM6DSL WHO_AM_I register */
    LSM6DSL_ReadID(&MotionSensor, &id);
    if (id != LSM6DSL_ID) {
        Error_Handler();
    }

    /* Initialize the LSM6DSL sensor */
    LSM6DSL_Init(&MotionSensor);

    /* Configure the LSM6DSL accelerometer (ODR, scale and interrupt) */
    LSM6DSL_ACC_SetOutputDataRate(&MotionSensor, 26.0f); /* 26 Hz */
    LSM6DSL_ACC_SetFullScale(&MotionSensor, 1); /* 1-1000mg; +1000mg */
    LSM6DSL_ACC_Set_INT1_DRDY(&MotionSensor, ENABLE); /* Enable DRDY */
    LSM6DSL_ACC_GetAxesRaw(&MotionSensor, &axes); /* Clear DRDY */

    /* Start the LSM6DSL accelerometer */
    LSM6DSL_ACC_Enable(&MotionSensor);
}
```

Figura 5. 2 - MEMs Init

Con questa funzione, colleghiamo i principali address che ci servono per l'accelerometro e lo configuriamo in modo da lavorare ad *1G*, in modo da ridurre l'incertezza relativa.

Procediamo inizializzando il sensore Wi-Fi, utilizzando le funzioni della libreria precedentemente integrata.

```
/* Wifi */  
WIFI_Status_t wstatus= WIFI_Init();
```

Figura 5. 3 - Wifi Init

```
WIFI_Status_t WIFI_Init(void)  
{  
    WIFI_Status_t ret = WIFI_STATUS_ERROR;  
  
    if(ES_WIFI_RegisterBusIO(&EsWifiObj,  
                            SPI_WIFI_Init,  
                            SPI_WIFI_DeInit,  
                            SPI_WIFI_Delay,  
                            SPI_WIFI_SendData,  
                            SPI_WIFI_ReceiveData) == ES_WIFI_STATUS_OK)  
    {  
        if(ES_WIFI_Init(&EsWifiObj) == ES_WIFI_STATUS_OK)  
        {  
            ret = WIFI_STATUS_OK;  
        }  
    }  
    return ret;  
}
```

Figura 5. 4 - Wifi Init, Corpo

Una volta inizializzato possiamo procedere con la connessione: dichiariamo le variabili di appoggio per identificare l'SSID e le passiamo alla funzione Wi-Fi Connect della libreria Wi-Fi.

```
/* USER CODE BEGIN PV */  
const char* NomeL = "Nome_SSID";  
const char* PasswL = "Psw_SSID";  
WIFI_Ecn_t ecn1=3;
```

Figura 5. 5 - Variabili Wi-Fi

```
WIFI_Connect(NomeL, PasswL, ecn1);
```

Figura 5. 6 - Richiamo della Wi-Fi Connect

```

WIFI_Status_t WIFI_Connect(
    const char* SSID,
    const char* Password,
    WIFI_Ecn_t ecn)
{
    WIFI_Status_t ret = WIFI_STATUS_ERROR;

    if(ES_WIFI_Connect(&EsWifiObj, SSID, Password, (ES_WIFI_SecurityType_t) ecn) == ES_WIFI_STATUS_OK)
    {
        if(ES_WIFI_GetNetworkSettings(&EsWifiObj) == ES_WIFI_STATUS_OK)
        {
            ret = WIFI_STATUS_OK;
        }
    }
    return ret;
}

```

Figura 5. 10 - Wi-Fi Connect

Ora che abbiamo la connessione Wi-Fi, possiamo procedere con la inizializzazione della MQTT, la connessione e il publish dei dati verso il server.

```

MQTTConnectInfo_t pConnectInfo;
pConnectInfo.cleanSession = true;
pConnectInfo.keepAliveSeconds = 60;
pConnectInfo.pClientIdentifier = "clientId-1ByEumwrr2";
pConnectInfo.clientIdentifierLength = strlen(pConnectInfo.pClientIdentifier);
pConnectInfo.pUserName = "UserNameClient";
pConnectInfo.userNameLength = strlen(pConnectInfo.pUserName);
pConnectInfo.pPassword = "PSSWclient";
pConnectInfo.passwordLength = strlen(pConnectInfo.pPassword);

mqttstatus = MQTT_Init(&pContext, &pTransportInterface, HAL_GetTick, userCallback, &pNetworkBuffer);

if(mqttstatus == MQTTSuccess)
    printf("\n\n Init Done. \n");
else
    printf("\n\n Init NOT Done. \n");

```

Figura 5. 9 - MQTT Init

```

MQTTPublishInfo_t* pWillInfo = NULL;
uint32_t timeoutMs=100;
bool* pSessionPresent=false;
mqttstatus = MQTT_Connect(&pContext, &pConnectInfo, pWillInfo, timeoutMs, pSessionPresent);

if(mqttstatus == MQTTSuccess)
    printf("\n\n Connect done. \n");
else
    printf("\n\n Connect NOT done. \n");

```

Figura 5. 8 - MQTT Connect

```

MQTTPublishInfo_t pPublishInfo;
uint16_t packetId;
LSM6DSL_Axes_t acc_axes;

// QoS of publish.
pPublishInfo.qos = MQTTQoS1;
pPublishInfo.pTopicName = "/some/topic/name";
pPublishInfo.topicNameLength = strlen( pPublishInfo.pTopicName );
pPublishInfo.pPayload = *acc_axes;
pPublishInfo.payloadLength = sizeof(LSM6DSL_Axes_t);

/* Timer2 */
HAL_TIM_Base_Start_IT(&htim2);

```

Figura 5. 7 - Payload + Start di Tim2

Per effettuare la publish abbiamo utilizzato il Timer2, configurato come illustrato nel capitolo 3, in modo da temporizzare gli invii.

```
void TIM2_IRQHandler(void)
{
    /* USER CODE BEGIN TIM2_IRQn 0 */

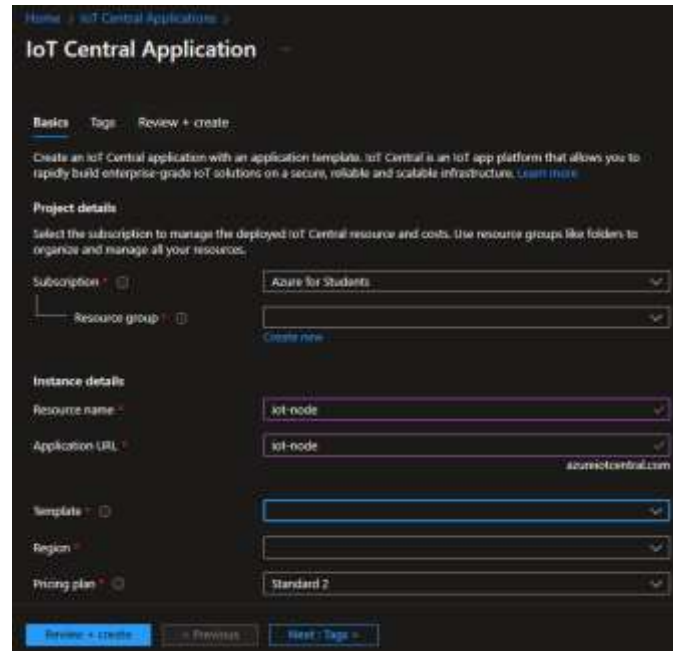
    /* USER CODE END TIM2_IRQn 0 */
    HAL_TIM_IRQHandler(&htim2);
    /* USER CODE BEGIN TIM2_IRQn 1 */
    // Packet ID is needed for QoS > 0.
    packetId = MQTT_GetPacketId( spContext );
    mqttstatus = MQTT_Publish(spContext, spPublishInfo, packetId);
    if( mqttstatus == MQTTSuccess ){
        while(pPublishInfo.qos) { // Since the qos is > 0, we will need to call MQTT_ReceiveLoop()
            MQTT_ReceiveLoop(spContext, timeoutMs);
        }
        // or MQTT_ProcessLoop() to process the publish acknowledgments.
    }
    /* USER CODE END TIM2_IRQn 1 */
}
```

Figura 5. 11 - Publish temporizzato

5.2 Dashboard di Azure

5.2.1 Creazione del device

Iniziamo creando il nostro IoT Central Application.



The screenshot shows the 'IoT Central Application' creation wizard in the Azure portal. The interface is in Italian. At the top, there's a breadcrumb 'Home > IoT Central Applications >'. Below it, the title 'IoT Central Application' is followed by tabs: 'Basics', 'Tags', and 'Review + create'. A descriptive paragraph explains that IoT Central is an IoT app platform for building enterprise-grade IoT solutions. The 'Project details' section asks for a subscription and resource group. The 'Subscription' dropdown is set to 'Azure for Students'. The 'Resource group' dropdown is empty, with a 'Create new' link below it. The 'Instance details' section includes 'Resource name' (set to 'iot-node'), 'Application URL' (set to 'iot-node' with 'azureiotcentral.com' as a hint), 'Template' (empty), 'Region' (empty), and 'Pricing plan' (set to 'Standard 2'). At the bottom, there are three buttons: 'Review + create' (highlighted in blue), '< Previous', and 'Next > Tags'.

Figura 5. 12 - IoT Central Application

Procediamo aprendo il nostro IoT Central Application, tramite l'Application URL, e apriamo la scheda 'Dispositivi'.



Figura 5. 13 - Dispositivi

Ora bisogna cliccare su '+ Nuovo' e procedere alla creazione del nostro primo dispositivo scegliendo con cura il nome del dispositivo e il suo ID.

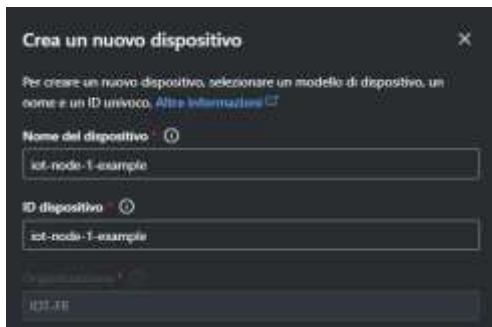


Figura 5. 15 - Crea un nuovo dispositivo pt.1

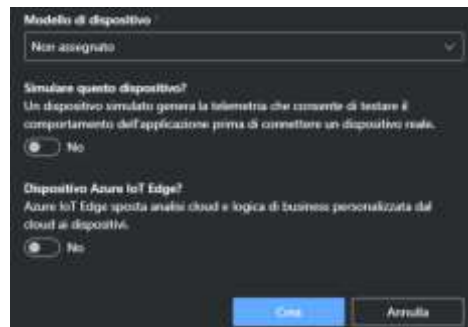


Figura 5. 14 - Crea un nuovo dispositivo pt.2

Una volta creato il dispositivo procediamo con la connessione tramite il tasto ‘connetti’ dove troviamo le chiavi necessarie.

5.2.2 Creazione della Dashboard

Una volta creato il dispositivo e aver effettuato la connessione, possiamo entrare nella scheda ‘Panoramica’ dove sono presenti tutti i dati che riceviamo dal nostro dispositivo.

Assicurata la ricezione dei dati, ci spostiamo nella scheda ‘Modelli Dispositivi’ e creiamo un nuovo modello: qui abbiamo due scelte, possiamo ricercare, tra i modelli già esistenti, quello adatto alla nostra board oppure procedere con la creazione di un nuovo modello, cliccando su ‘Dispositivo IoT’.

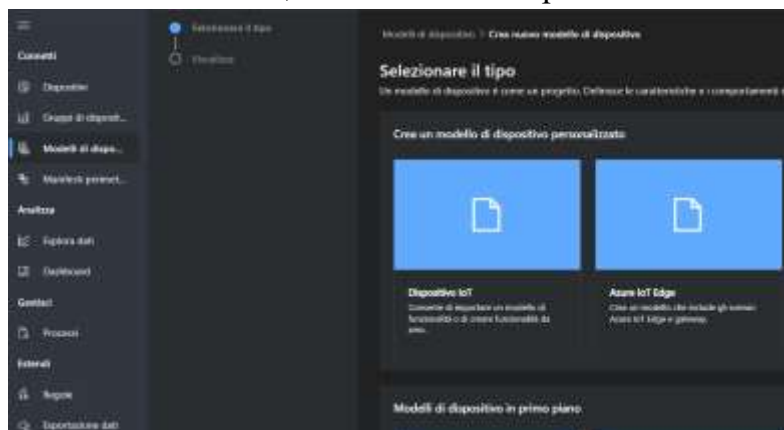


Figura 5. 16 - Modello Dispositivo IoT

Una volta scelto il nome, finalizziamo con la creazione del modello del dispositivo. Ora dobbiamo creare il modello da visualizzare: anche qui possiamo importare uno già esistente o crearlo da zero. Procediamo creandolo da zero, selezionando ‘Modello Personalizzato’ e aggiungendo le funzionalità necessarie al nostro progetto.

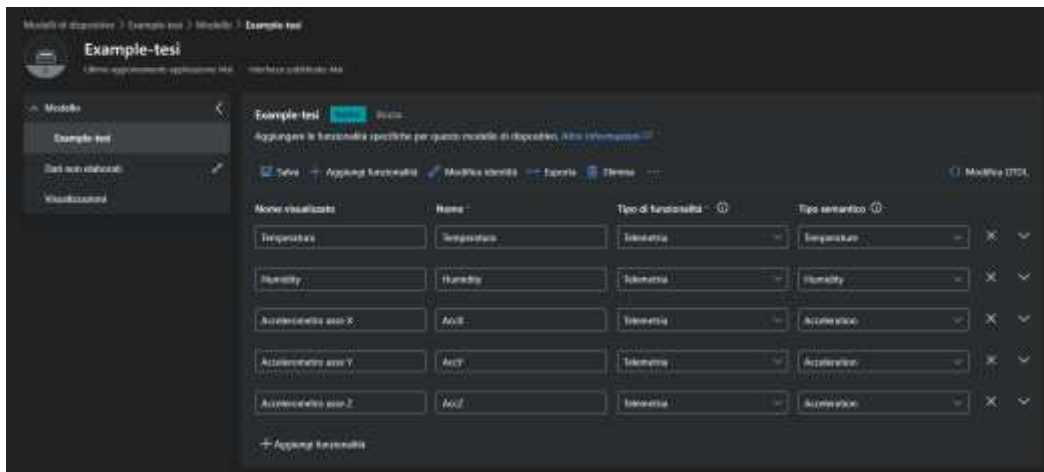


Figura 5. 18 - Modello Personalizzato

Salviamo e spostiamoci nella scheda ‘Visualizzazioni’, selezioniamo ‘Visualizzazione del dispositivo’. Da qui possiamo procedere alla creazione del nostro modello di visualizzazione del dispositivo in utilizzo: trasciniamo i widget presenti sulla sinistra della schermata e procediamo alla personalizzazione, cliccando sul simbolo a forma di ‘Matita’ sul widget scelto.

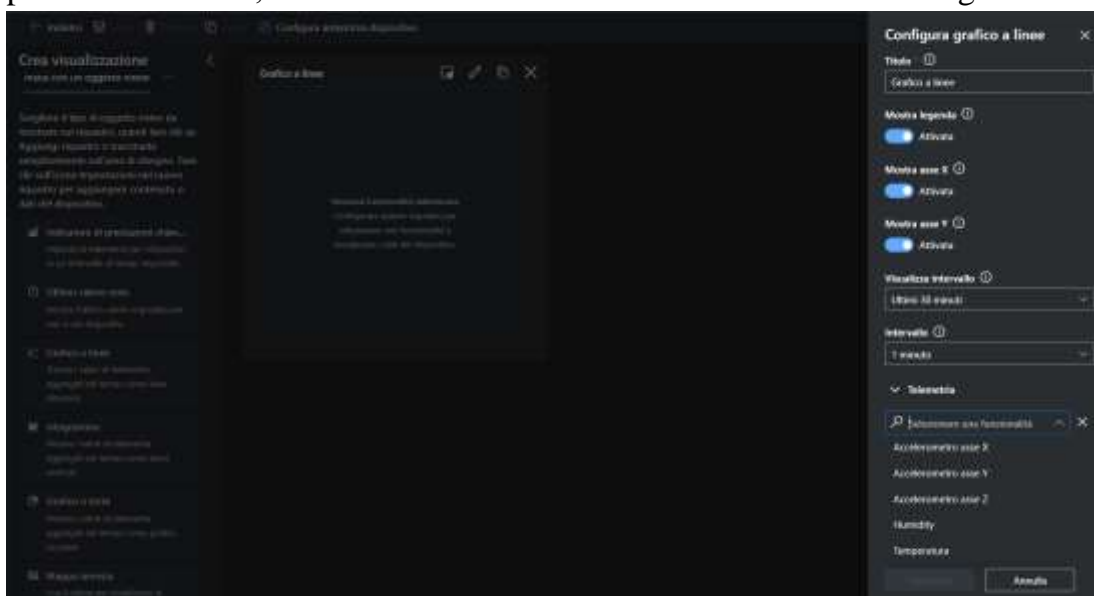


Figura 5. 17 - Personalizzazione Widget

Scegliamo la telemetria che vogliamo visualizzare e clicchiamo su ‘Aggiorna’. Crea la nostra visualizzazione, colleghiamola al dispositivo che vogliamo tramite il tasto ‘Configura anteprima dispositivo’, dove è presente una lista dei nostri dispositivi. Una volta selezionato il nostro widget comincerà a visualizzare i dati scelti.

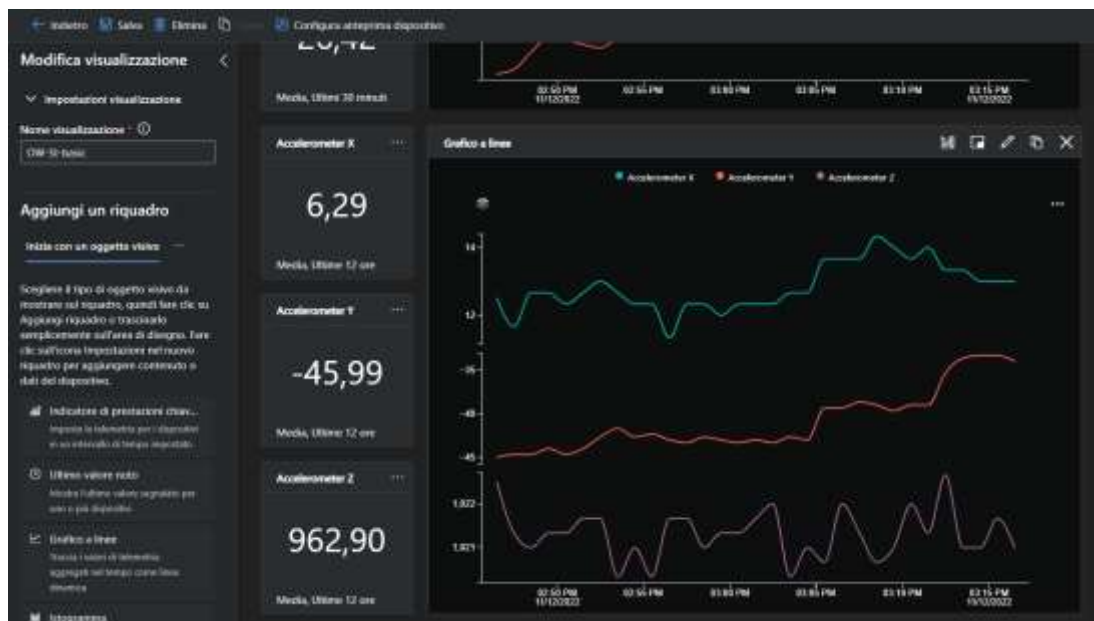


Figura 5. 20 - Visualizzazione finita

Salvato il nostro modello di visualizzazione per la tipologia del nostro dispositivo, reiteriamo il procedimento per la scheda 'Dashboard', dove possiamo creare un'interfaccia personalizzata per la visualizzazione di più dispositivi contemporaneamente:



Figura 5. 19 - Dashboard

Capitolo 6 – Sviluppi Futuri

6.1 FFT e Wavelet Transform

Una possibile implementazione, per migliorare l'analisi dei dati generati dai sensori, è l'introduzione degli algoritmi di Fast Fourier Transform e della Wavelet Transform. Questi algoritmi possono essere eseguiti direttamente sul nodo sensore al fine di migliorare i dati raccolti.

Nel caso dell'algoritmo di FFT, analizzando i dati nel dominio della frequenza, provenienti dall'accelerometro, possiamo ispezionare come il soggetto dell'analisi reagisce ad ogni frequenza. Ciò significa che possiamo ottenere gli spettri di frequenza utili al fine di effettuare ottimizzazioni del progetto.

6.2 AI – Forecasting

Un'ulteriore possibilità di sviluppo è l'introduzione di un sistema automatizzato che analizza i dati in ingresso, provenienti da ogni dispositivo predisposto, e che sia capace di comprenderli. Ovviamente ci avvaliamo dell'ausilio degli algoritmi di intelligenza artificiale che sono capaci di apprendere in modo autonomo grazie all'elevata mole di dati disponibili. Questo sistema automatizzato ci serve al fine di effettuare un'analisi più veloce dei dati per ricevere, quando la nostra AI sarà effettivamente addestrata, delle previsioni estremamente realistiche sulle condizioni in cui versa l'oggetto che stiamo analizzando.

6.3 Deep Learning Algorithms

Un'importante sviluppo, a fini di migliorare la sicurezza e, di conseguenza, la robustezza del progetto, riguarda l'utilizzo di algoritmi di Deep Learning, come il Convolution Neural Network (CNN) oppure Long Short Term Memory (LSTM), per l'identificazione degli attacchi IoT negli IoT Frameworks. Questi attacchi avvengono spesso utilizzando la tecnica del Man-in-the-middle oppure del Denial of Service, con lo scopo di accedere ai dati sensibili e privati. Questi algoritmi giocano un ruolo importante nella predizione degli attacchi grazie ad un'alta accuratezza. In futuro questi modelli potranno essere utilizzati come firewall per predire nodi malevoli all'interno dell'ambiente IoT.

Bibliografia

- Inventek System. (n.d.). *ISM43362-M3G-L44-E/U Serial-to-WiFi Module*. Retrieved from .inventeksys: <https://www.inventeksys.com/ism4336-m3g-l44-e-embedded-serial-to-wifi-module/>
- Mayank Mishra, P. B. (2022). Structural health monitoring of civil engineering structures by using the internet of things: A review. *Journal of Building Engineering*.
- MQTT. (n.d.). *MQTT 3.1 Specification*. Retrieved from mqtt: <https://public.dhe.ibm.com/software/dw/webservices/ws-mqtt/mqtt-v3r1.html>
- ST. (n.d.). *Capacitive digital sensor for relative humidity and temperature*. Retrieved from st: <https://www.st.com/en/mems-and-sensors/hts221.html>
- ST. (n.d.). *iNEMO 6DoF inertial measurement unit (IMU), for smart phones and battery operated IoT, Gaming, Wearable and Consumer Electronics. Ultra-low power and high accuracy*. Retrieved from st: <https://www.st.com/en/mems-and-sensors/lsm6dsl.html/>
- ST. (n.d.). *STM32Cube*. Retrieved from st: <https://www.st.com/en/ecosystems/stm32cube.html>
- ST. (n.d.). *STM32L4 Discovery kit IoT node, low-power wireless, BLE, NFC, SubGHz, Wi-Fi*. Retrieved from www.st.com: <https://www.st.com/en/evaluation-tools/b-l475e-iot01a.html>

INDICE DELLE FIGURE

FIGURA 1. 1 - ARCHITETTURE IOT PER APPLICAZIONI SHM.....	6
FIGURA 2. 1 - SCHEMA DELLA BOARD.....	8
FIGURA 2. 2 - HARDWARE BLOCK DIAGRAM.....	9
FIGURA 2. 3 - TOP VIEW DELLA BOARD	11
FIGURA 2. 4 - LSM6DSL.....	12
FIGURA 2. 5 - SENSITIVITY ACC + GYRO	12
FIGURA 2. 6 - DEVICE SUMMARY	12
FIGURA 2. 7 - HTS221	13
FIGURA 2. 8 - SENSITIVITY HTS221	13
FIGURA 2. 9 - BETTER DISTANCE.....	14
FIGURA 2. 10 - ISM43362-M3G-L44.....	15
FIGURA 3. 1 - BOARD SELECTION.....	16
FIGURA 3. 2 - CLOCK CONFIGURATION	17
FIGURA 3. 3 - PINOUT & CONFIGURATION.....	18
FIGURA 3. 4 - SOFTWARE PACKS COMPONENT SELECTOR.....	18
FIGURA 3. 5 - SOFTWARE PACKAGE MEMS1	19
FIGURA 3. 6 - I2C CONFIGURATION.....	20
FIGURA 3. 7 - SPI3 CONFIGURATION.....	20
FIGURA 3. 8 - TIM2 CONFIGURATION	21
FIGURA 3. 9 - PANORAMICA IDE.....	22
FIGURA 4. 1 - SCHERMATA ALL SERVICES DI MICROSOFT AZURE	24
FIGURA 4. 2 - SCHEMA MQTT	25
FIGURA 4. 3 - FIXED HEADER.....	26
FIGURA 4. 4 - DASHBOARD EXAMPLE.....	29
FIGURA 5. 1 - INIZIALIZZAZIONE.....	30
FIGURA 5. 2 - MEMS INIT.....	30
FIGURA 5. 3 - WIFI INIT	31
FIGURA 5. 4 - WIFI INIT, CORPO	31
FIGURA 5. 5 - VARIABILI WI-FI.....	31
FIGURA 5. 6 - RICHIAMO DELLA WI-FI CONNECT	31
FIGURA 5. 7 - PAYLOAD + START DI TIM2.....	32

FIGURA 5. 8 - MQTT CONNECT	32
FIGURA 5. 9 - MQTT INIT	32
FIGURA 5. 10 - WI-FI CONNECT.....	32
FIGURA 5. 11 - PUBLISH TEMPORIZZATO	33
FIGURA 5. 12 - IOT CENTRAL APPLICATION	34
FIGURA 5. 13 - DISPOSITIVI	34
FIGURA 5. 14 - CREA UN NUOVO DISPOSITIVO PT.2	35
FIGURA 5. 15 - CREA UN NUOVO DISPOSITIVO PT.1	35
FIGURA 5. 16 - MODELLO DISPOSITIVO IOT.....	35
FIGURA 5. 17 - PERSONALIZZAZIONE WIDGET	36
FIGURA 5. 18 - MODELLO PERSONALIZZATO.....	36
FIGURA 5. 19 - DASHBOARD.....	37
FIGURA 5. 20 - VISUALIZZAZIONE FINITA.....	37