Applicazioni di *Internet delle Cose* basate sull'impiego del *System on Chip* (SoC) ESP-8266

Sebbene *Internet* sia formalmente una *rete di computer* questi non sono che il tramite per mettere in comunicazione gli utenti che li usano, offrendo tutta una serie di servizi che vanno dalla posta elettronica alla fruizione di contenuti multimediali per non parlare del commercio elettronico.

Internet delle cose si propone invece di utilizzare l'infrastruttura già esistente di Internet per mettere in comunicazione una miriade di dispositivi elettronici di vario tipo, allo scopo di controllarli a distanza ed in prospettiva creare degli *ambienti intelligenti*.
Gli oggetti presi inconsiderazione sono i più disparati: alcuni molto grandi come le Jeep (funzioni di monitoraggio e manutenzione remota) e i frigoriferi (che alcuni vorrebbero dotare della capacità di "fare la spesa" riordinando automaticamente i cibi man mano che vengono consumati), altri estremamente piccoli come i microchip inseriti nei passaporti (allo scopo di garantire l'autenticità dei documenti), nei titoli di viaggio (apertura automatica dei tornelli di accesso ai binari) e nelle etichette dei prodotti in vendita nei supermercati (funzioni anti taccheggio e tracciabilità).

Chiaramente che ci sono dei grossi interessi al riguardo: i governi e tutte le grandi istituzioni si aspettano di ottenere dei benefici dalla possibilità di monitorare più efficacemente il territorio e i cittadini per non parlare delle applicazioni logistiche come la spedizione delle merci, la gestione dei magazzini, sino alla possibilità di eliminare le cassiere sostituendole con dei portali capaci di fare automaticamente l'inventario delle confezioni presenti nel carrello interrogando i *tag* RFID dei singoli prodotti.
Date le premesse si tratta di vedere se da questa evoluzione che pare inevitabile possano venire dei benefici anche per gli utenti finali (ovvero per i cittadini/consumatori).

Da anni, ad esempio, si parla di **domotica** cioè della possibilità di realizzare delle case intelligenti, tuttavia fatto salvo per i grandi edifici le suddette tecnologie non hanno sinora incontrato un grande successo, in parte per il costo elevato degli apparecchi ma anche e sopratutto per la complessità di gestione che impone di rivolgersi a dei professionisti o perlomeno di seguire dei corsi.

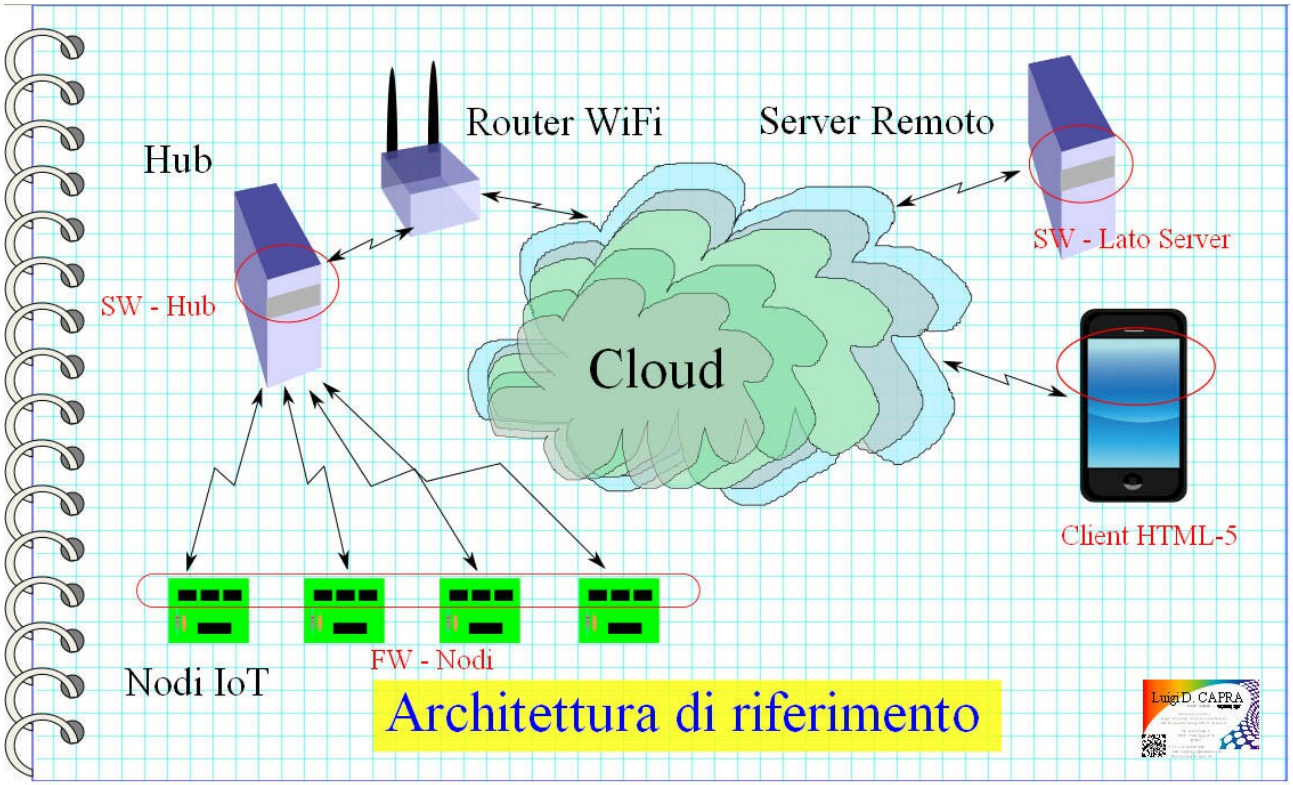
Oggi tutti i grandi produttori di hardware e di software (da Philips a Google) stanno tornando alla carica su questa questione proponendo lampadine intelligenti, termostati controllati a distanza e quant'altro.

Ancora una volta non si può fare a meno di rilevare come la curiosità dei consumatori sia ostacolata dai prezzi elevati dei prodotti offerti sugli scaffali.
Ad esempio la Philips per 200€ offre un kit costituito da tre lampadine intelligenti e una centralina.

Inutile dire che la gente ci pensa due volte prima di fare degli acquisti con il risultato di raffreddare gli entusiasmi dei produttori ed in particolare delle PMI che potrebbero essere eventualmente interessate ad offrire ai clienti la possibilità di controllare in remoto i loro prodotti.

In realtà si tratta della calma prima della tempesta perché i prezzi delle soluzioni di *Internet of Things* (**IoT**) sono in caduta libera.
Già oggi si possono acquistare delle schede IoT equipaggiate di un relè e di un processore potente come un computer degli anni '90, capace di connettersi ad Internet tramite WiFi, per la modica cifra di 5,31€. Vale a dire meno del costo della lampadina che dovrebbe controllare!

Per realizzare delle applicazioni non manca che il software.
I protocolli di Internet possono essere facilmente immaginati come una sorta di “collante” avente lo scopo di garantire interoperabilità all'interno di un coacervo di applicazioni software e sistemi hardware estremamente eterogenei.
Tale aspetto è ancor più evidente nel contesto di *Internet of Things* (IoT), al punto che parlando di software per Internet delle Cose è necessario introdurre una sorta di “geografia” allo scopo di caratterizzare ubicazione dell'applicazione considerata nel quadro generale.



In questo contesto si individuano quattro diverse tipologie di applicazioni IoT:

1) il *firmware* utilizzato per far funzionare i nodi cioè monitorare e/o controllare i dispositivi.
2) il software di gestione delle centraline (*hub*) che supervisionano i nodi, gestendo le comunicazioni verso la rete.
3) i *server remoti* utilizzati per la memorizzazione dei dati rilevati e l'elaborazione statistica degli stessi.
4) I *web-client* ovvero le interfacce che consentono agli utenti di interagire con i nodi da remoto.

La tassonomia raffigurata si giustifica osservando che le quattro tipologie di applicazione presuppongono l'impiego di piattaforme hardware e di linguaggi di programmazione differenti.

I *nodi* impiegati per monitorare i sensori ed inviare i comandi agli attuatori si basano tipicamente sull'impiego di un *System on Chip* (SoC) come l'Espressif ESP-8266 o di una piattaforma di elaborazione a basso costo.
Il ruolo di *centralina* può essere affidato a seconda dei casi ad un PC o ad una scheda di elaborazione un po' più potente di quelle dei nodi, su cui girano una serie di software che offrono vari tipi di servizi ai nodi (*firewall*, *storage*, *time-date*, eccetera).
Il ricorso a *server remoti* ospitati nelle strutture di un fornitore di servizi di *hosting* si giustifica con

l'esigenza di garantire la conservazione dei dati raccolti anche nel caso in cui i nodi e la centralina fossero oggetto di atti vandalici, nonché la possibilità (attraverso la duplicazione delle stazioni di *storage*) di consultare le informazioni anche nel caso in cui la connessione con la località dove si trovano i nodi dovesse interrotta.
Per quanto concerne il *pannellino di controllo* risulta infine conveniente sviluppare l'interfaccia utente impiegando le normali tecnologie *web*. Il ricorso a pagine HTML-5 + JavaScript assicura l'assoluta indipendenza dalla piattaforma hardware e al sistema operativo, garantendo la possibilità di fruire dei dati utilizzando sistemi così diversi come un PC Windows o Linux, un iPad o uno smart-phone Android.

Proposte

In riferimento al quadro precedente la mia offerta comprende:

Funzioni software operanti a livello di *nodo.*

1) **Console per il controllo remoto dell'ESP-8266 tramite WiFi**
In passato molti sviluppatori IoT hanno scelto di congelare i programmi di controllo dei loro nodi sotto forma di *firmware*. Una soluzione apparentemente ottimale che però espone a dei seri rischi nel caso in cui dovessero cambiare le condizioni operative. L'esigenza di aggiornare il firmware potrebbe infatti implicare la necessità i rimuovere le schede per accedere alle memorie da riprogrammare.
Ipotizzando una vita operativa dell'ordine della decina d'anni non si può trascurare la probabilità che nascano delle nuove esigenze, nel qual caso sarebbe utile disporre della possibilità di effettuare una diagnostica remota e poter improvvisare delle soluzioni, modificando le dinamiche di funzionamento dei nodi, senza essere obbligati a smontarli, fatto che giustifica l'utilità della Console WiFi.
L'applicazione considerata presuppone l'esistenza di un *client* (la **Console** vera è propria) ed un *server* sovraintende al *nodo* monitorandone lo stato e controllando l'esecuzione dei comandi ricevuti dalla **Console** remota.
Nello specifico la **Console** consiste in una pagina *web*, scritta in HTML-5 + JavaScript e quindi fruibile per mezzo di un browser su di una qualsiasi piattaforma (PC, *tablet*, *smart-phone*).
L'interfaccia utente offre tre funzionalità: pannello di controllo, terminale e generatore di codice.
Ispirato al LuaLoader di NodeMCU, il pannellino di controllo della Console WiFi è il principale strumento di diagnostica a disposizione dell'operatore, che si trova ad avere letteralmente tutte le funzioni di monitoraggio e i controlli a portata di click.
In seconda battuta, preso atto delle mutate esigenze, l'operatore potrebbe improvvisare una soluzione operando in maniera interattiva avvalendosi della possibilità di inviare dei comandi Lua all'interprete che risiede a bordo del nodo.
Ottenuto un risultato significativo la soluzione potrebbe essere caricata via WiFi per essere salvata in modo permanente nel filesystem dell'ESP-8266.
A coronamento dell'offerta occorre ricordare la possibilità per gli utenti di configurare i nodi in maniera semi-automatica avvalendosi di un apposito generatore di codice. Per configurare i nodi non è pertanto necessario scrivere del codice ma basta agire sui controlli di un pannellino, compilando l'equivalente di un modulo prestampato (*form*). Al termine il programma convertirà le impostazioni in righe di codice Lua e le caricherà automaticamente nella memoria dell'ESP-8266 pronte per essere provate.

Non bisogna infine dimenticare che la disponibilità di una Console WiFi “libera” la porta seriale RS232 dell'ESP-8266 (normalmente impiegata dai terminali utilizzati per interagire con l'interprete Lua) rendendola disponibile per altre applicazioni (come la seguente).

2) Software utilizzabile per realizzare un **Bridge WiFI-RS232**.
Il software in questione consente di controllare dei dispositivi remoti, equipaggiati di porta RS232, utilizzando un cavo RS232 “virtuale” realizzato utilizzando ad un estremo la porta RS232 in dotazione del System on Chip ESP-8266 dall'altra le connessioni WiFi e /o Internet per collegarsi ad un sistema remoto.
NB: la fattibilità dell'applicazione in relazione ad eventuali requisiti di real-time deve essere esaminata caso per caso.

3) Software per il caricamento di dati telemetrici su di un sito remoto.

4) Software per il controllo di sensori e/o attuatori costituiti da relè.

Funzioni software operanti a livello di *centralina.*

5) Mini server REST, *platform-independent*, sviluppato in C-Language.
Piattaforma Intel x86: supporto Windows XP, 7, 8, 10, Linux (Ubuntu, Open Mandriva, altri).
Piattaforma Raspberry Pi. Sistema operativo Rapsbian.
Server HTML+HTTP. Funzioni REST. Possibilità di interfacciarsi all'hardware locale per mezzo di funzioni scritte in C o in Assembler allo scopo di interfacciarsi a varie tipologie di dispositivi.

6) Soluzioni personalizzate per il salvataggio dei dati raccolti in un database basate sull'utilizzo di un server XAMPP (Apache + MySQL + PHP).

Funzioni software operanti sul server di un *Hosting Provider.*

7) Soluzioni personalizzate per il salvataggio dei dati raccolti in un database basate sull'utilizzo di un server XAMPP (Apache + MySQL + PHP).
Elaborazioni statistiche.
Fusione dei dati in un quadro sintetico personalizzato.
Gestione tracciabilità.
Supporto monitoraggio della Qualità.

Pannellini di controllo.

8) Possibilità di sviluppare delle *pagine web* dinamiche o statiche in HTML-5 + CSS-3 + JavaScript utilizzabili come pannellini di controllo (*platform-independent*) per il monitoraggio.

Sistemi industriali

Alcune delle tecnologie descritte in precedenza si prestano ad essere utilizzate in abbinamento a

sistemi già esistenti allo scopo di “metterli in rete”. Tipicamente si tratta di sviluppare un server REST e di realizzare un pannellino di controllo remoto.

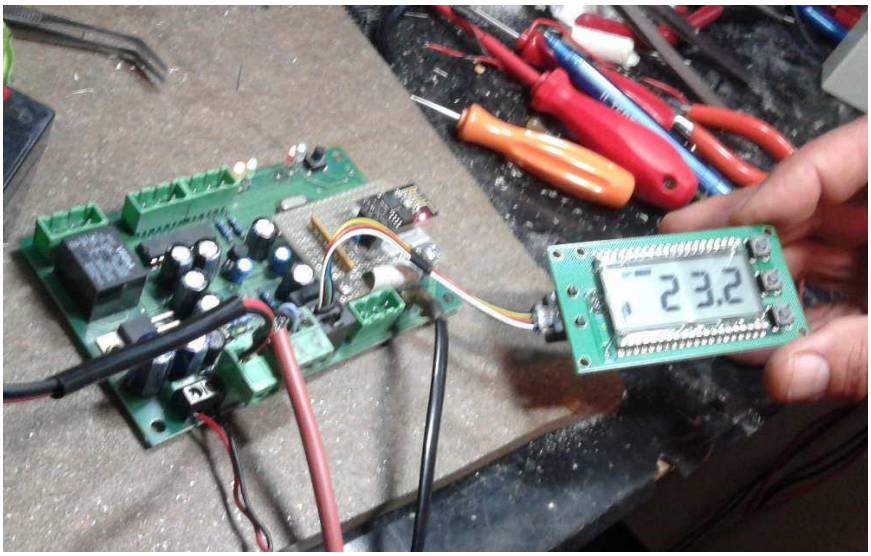
Consulenza

Offro la mia disponibilità a coadiuvare i clienti nello sviluppo di applicazioni basate sull'impiego dei componenti software elencati in precedenza.

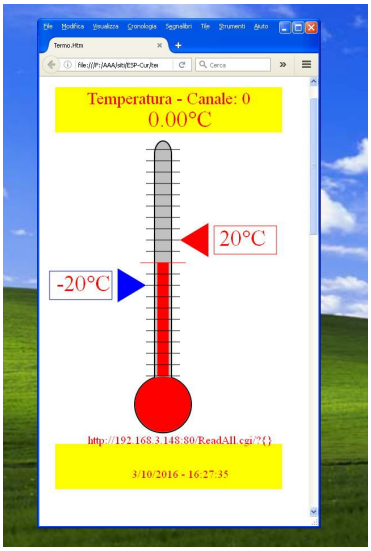
Case studies

Oltre alla Console WiFi, descritta in precedenza, sono state sviluppate diverse applicazioni di prova.

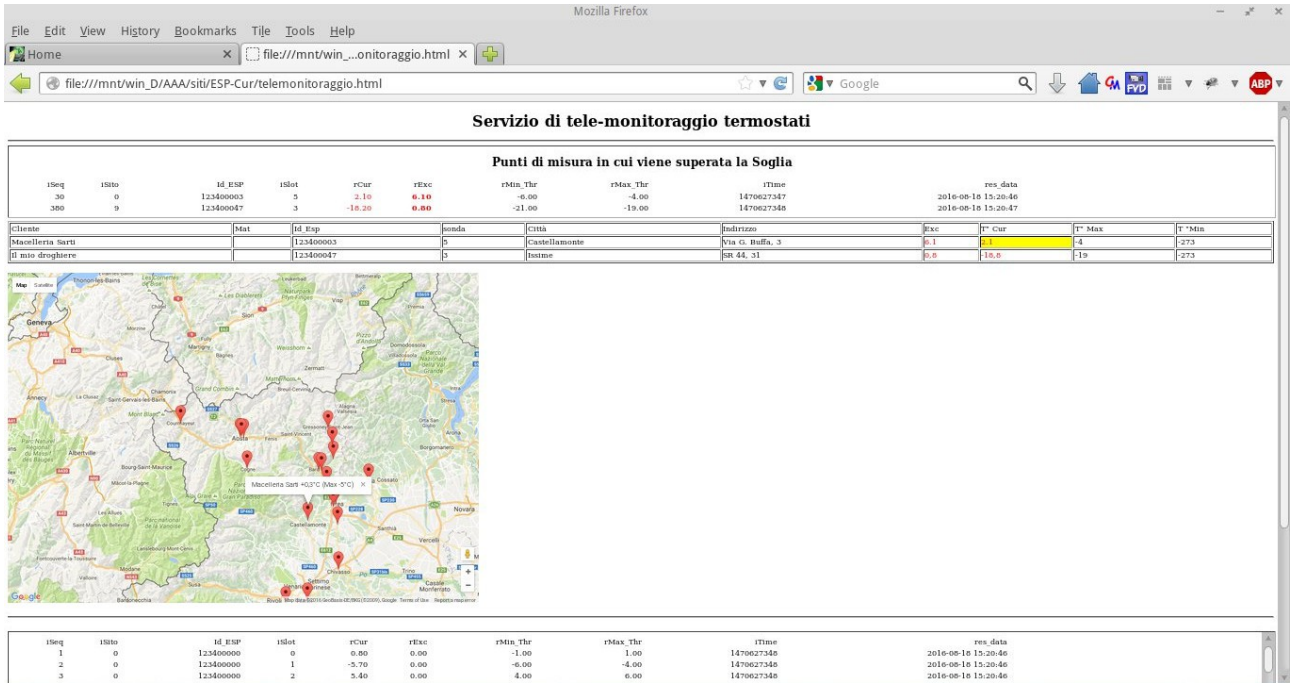
Termostato controllato a distanza



Per verificare l'affidabilità della tecnologia proposta ho utilizzato un SoC ESP-8266 allo scopo di controllare a distanza, tramite Internet, un termostato già esistente.
L'esperimento prevede che ogni due minuti le letture del termometro vengano inviate ad un applicazione PHP, residente sul sito di un Hosting Provider per la registrazione in un database.
Le letture più recenti possono essere consultate cliccando sul seguente link:
http://www.luigidcapra.com/prove/esp_get.php



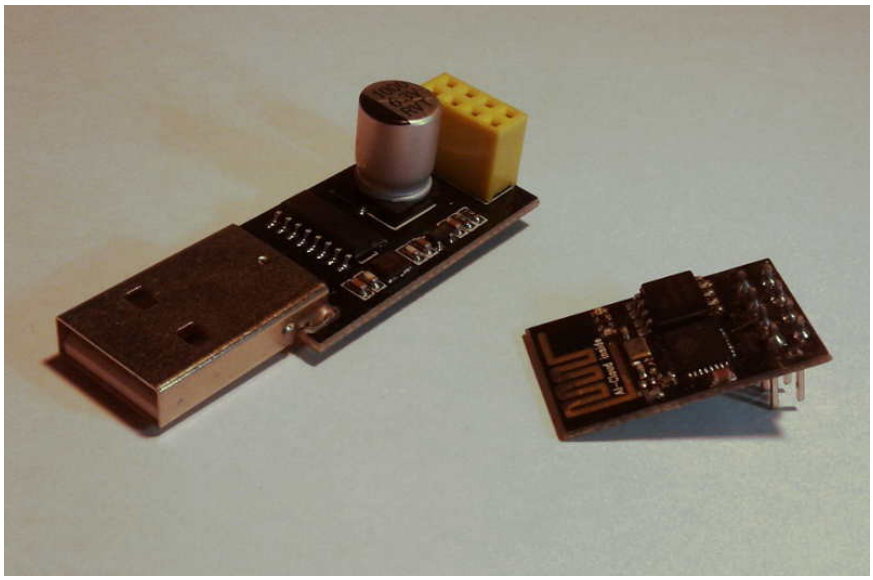
Monitoraggio della temperatura di celle frigorifere



Ampliando l'esperimento precedente si è ipotizzato di gestire un migliaio di punti di misura ubicati in una ventina di siti diversi sparsi sul territorio (corrispondenti ad un centinaio di nodi ESP-8266).
Ciascun nodo comunica le sue letture al server remoto con una cadenza prestabilita (ad esempio ogni 15 minuti).
Alla ricezione di ogni pacchetta di dati l'applicazione PHP che gestisce il database verifica il rispetto dei vincoli prefissati. Nel caso in cui si rilevi una discrepanza il programma di analisi avviserà il responsabile della cella frigorifera inviandogli una email.
I dati rilevati saranno quindi inseriti in un quadro sintetico (tabulato riassuntivo) evidenziando i punti di misura dove si è verificata una violazione dei limiti.
Infine, i dati opportunamente geolocalizzati verranno presentati graficamente sotto forma di bandierine (che saltano) sovrapposte ad una carta di Google Map, il tutto allo scopo di facilitare i tecnici incaricati della manutenzione delle celle frigorifere nella pianificazione del percorso da seguire nelle loro visite.

Cavo Seriale virtuale

In virtù della facilità con cui può essere integrata, l'interfaccia seriale RS232 continua ad essere una dotazione standard di quasi tutti gli strumenti di misura, ad onta di ogni tentativo dei fabbricanti di PC di eliminarla.
Circostanza che crea non pochi problemi quando occorre collegare un PC ad uno strumento di misura poiché in mancanza di porte seriali si è costretti ad a ricorrere a delle schedine di conversione dal funzionamento non sempre perfetto. In un simile contesto, il ricorso alle funzionalità di comunicazione del system on chip ESP-8266 può essere una valida alternativa all'utilizzo dei classici convertitori USB-RS232.
Poiché il System on Chip ESP-8266 dispone sia di una porta RS232 che di WiFi si può ipotizzare di interfacciarsi allo strumento di misura per mezzo della seriale e di utilizzare quindi una connessione TCP/IP realizzata per mezzo del WiFi per comunicare con il PC; ma c'è di più poiché la disponibilità di uno *stack* TCP/IP consente di collegarsi ad Internet (tramite un *modem/route*r WiFI) prospettando la possibilità (se non ci sono vincoli di *real-time*) di controllare da remoto lo strumento.



Per realizzare il tutto non occorre che un ESP-8266 e uno schedino di interfaccia USB-RS232 (preferibilmente già predisposto per l'interfacciamento all'ESP8266) e ovviamente un software sviluppato *ad hoc* per gestire le peculiarità dello strumento considerato.

Luigi D. CAPRA
капра луиджи - 路易吉
Software Consultant
Image Processing, Artificial Vision Systems,
Non Destructive Testing (NDT), Firmware.
Via della Chiesa, 6
10030 - Villareggia (TO)
ITALY
Tel. ++39 3312845208
e-mail: luigidcapra@katamail.com
http://www.luigidcapra.com