UNIVERSITÀ POLITECNICA DELLE MARCHE
FACOLTÀ DI INGEGNERIA

Corso di Laurea in Ingegneria Informatica e
dell'Automazione

Sviluppo di un'applicazione WPF per il monitoraggio da remoto di macchinari controllati da PLC

WPF application development for remote monitoring of PLC controlled machines

Relatore: Chiar.mo
Conte Giuseppe

Candidato:
Mezzanotte Matteo

A.A. 2019/2020

Sviluppo di un'applicazione WPF per il monitoraggio da remoto di macchinari controllati da PLC

Matteo Mezzanotte

Relatore prof. Giuseppe Conte

Università Politecnica delle Marche, © Dicembre 2020.

Abstract

Scopo di questa documentazione è esporre il lavoro svolto durante i mesi di ottobre e novembre 2020 all'interno dell'azienda Messersì Packaging di Barbara (AN). Il lavoro si è incentrato sulla realizzazione di un'applicazione in tecnologia WPF (e successivamente Xamarin.Forms per dispositivi Android) per la connessione, il monitoraggio e il controllo di sistemi automatizzati per l'imballaggio. Gli impianti in questioni sono comandati da PLC ai quali ci si connette tramite rete Internet (IP protocol). Il framework .NET ha fornito i mezzi necessari all'integrazione dell'applicativo con PLC di brand Siemens e database relazionali. Le fasi di lavoro che hanno condotto al risultato finale, le problematiche incontrate e gli ulteriori orizzonti di sviluppo sono ampiamente discussi nei capitoli che compongono la presente tesi.

Indice

Elenco delle figure

Elenco delle tabelle

Capitolo 1
Introduzione

Nell'ambito dell'automazione industriale, l'avvento delle nuove tecnologie e della cosiddetta quarta rivoluzione industriale hanno comportato notevoli innovazioni nelle tecniche di controllo dei sistemi produttivi.

Nella fattispecie, l'integrazione delle tecnologie di informazione e comunicazione digitale promuove una nuova visione del sistema manufatturiero, anche attraverso una nuova gestione della linea produttiva, caratterizzata dell'interconnessione di ogni entità della catena tramite Internet.

Le tecniche di monitoraggio da remoto hanno permesso ai produttori di implementare sistemi di telemanutenzione reattiva e predittiva, ma che allo stesso tempo permettono un'accurata analisi della qualità del prodotto e della conseguente soddisfazione del cliente.

Il progetto svolto all'interno dell'azienda Messersì Packaging, leader nella produzione di macchinari automatizzati per l'imballaggio, è nato proprio dall'esigenza di sviluppare queste nuove metodologie nell'ottica di un ammodernamento del sistema di assistenza e di monitoraggio dei macchinari prodotti.

A tale scopo è stata promossa la realizzazione di un sistema, denominato Messersì Supervisor, che consista in un applicativo in grado di connettersi tramite Internet ai PLC controllanti gli impianti produttivi. Questa connessione permetterà di avere l'informazione sullo stato di funzionamento dei macchinari in ogni momento e da qualsiasi posizione.

L'applicazione mira a fornire all'azienda uno strumento che integri i sistemi di manutenzione e assistenza, fornendo allo stesso tempo un flusso di informazioni proveniente dai sistemi già installati.

Messersì Supervisor basa il suo funzionamento sugli strumenti del framework Microsoft .NET e sulla connessione tramite protocollo IP a PLC di brand Siemens. Sono state realizzate due varianti dell'applicativo: la prima è una applicazione WPF per sistemi Windows, pensata quindi per l'utilizzo da parte del personale Messersì dalla sede aziendale. La seconda è invece una versione mobile per dispositivi Android, pensata per l'impiego sul posto da parte di un tecnico e in previsione di un futuro rilascio per il cliente.

L’elaborato è stato realizzato secondo la seguente struttura:

- *Il capitolo 2* illustra innanzitutto le specifiche del progetto e le fasi di lavoro portate avanti con l’azienda.
- *Il capitolo 3* fornisce una panoramica sulle innovazioni della quarta rivoluzione industriale e dell’Internet of Things, contesto nel quale si inserisce il sistema di interconnessione realizzato.
- *Il capitolo 4* contiene la trattazione delle caratteristiche tecniche di un PLC, controller industriale oggetto del monitoraggio.
- *Il capitolo 5* raccoglie gli strumenti e i software alla base di questo sistema.
- *il capitolo 6* descrive quindi gli obiettivi raggiunti e illustra il risultato finale del lavoro.
- *Il capitolo 7* tratta infine le modalità di svolgimento dei test condotti, oltre a una raccolta dei risultati raggiunti, dei problemi incontrati e degli orizzonti di miglioramento.

Capitolo 2
Specifiche e fasi di lavoro

In questo capito si intende illustrare nel dettaglio specifiche, modalità di realizzazione e considerazioni che hanno portato allo sviluppo del progetto Messersì Supervisor, ampiamente descritto nei capitoli di questa tesi.

2.1 Considerazioni preliminari

L'opportunità di sviluppare questo progetto è stata offerta dalla ditta Messersì Packaging di Barbara, in provincia di Ancona. L'azienda, fondata nel 1980 da Maurizio Messersì, è leader nel settore del packaging, con una produzione annua di più di 500 macchine reggiatrici, avvolgitrici, incappucciatrici e soluzioni varie per l'imballaggio.
Si tratta di una impresa altamente internazionalizzata, con sedi in Francia e Spagna e che destina il 70 % della produzione all'export in oltre 60 paesi sparsi nei 5 contenenti[1].

Figura 2.1 - Impianti Messersì nel mondo

Proprio dalla capillarità della rete di diffusione delle soluzioni Messersì è nata l'idea di un sistema che permettesse di ricavare l'informazione dello stato di funzionamento delle macchine in tempo reale ovunque esse si trovino.

[1] https://messersi.com/it/azienda/

È stato quindi pensato un software per la supervisione da remoto di macchinari e impianti, un applicativo in grado di connettersi, monitorare e controllare i sistemi produttivi tramite Internet. Inoltre, è stato richiesto di portare avanti lo sviluppo in funzione dell'installazione su smartphone/tablet e in ottica IoT. Infatti, il progetto dovrà essere precursore per lo sviluppo di sistemi di manutenzione predittiva e di miglioramento dell'assistenza al cliente.

2.2 Fasi progettuali

Il progetto ha preso il nome di Messersì Supervisor ed è stato portato avanti in collaborazione con l'ufficio tecnico della sede di Barbara.
Qui ci si è riuniti per definire le specifiche e valutare i progressi settimanalmente durante i mesi di sviluppo. L'azienda ha inoltre fornito gli strumenti necessari allo sviluppo, in particolare i PLC di test.
In primo luogo, è stata dunque definita la tabella di marcia necessaria a porre obiettivi e scadenze:

Punto	Descrizione
1	Definizione tabella generale per lo scambio dati per famiglia di macchine
2	Definizione struttura dell'applicazione da sviluppare su PC: home page e pagine secondarie di monitoraggio macchina, funzioni da implementare. Funzioni minime: 1. Pagina generale di verifica dei collegamenti macchine attivi 2. Pagina di diagnostica del collegamento (frame con messaggi di allarme o problemi di connessione) 3. Pagina di monitoraggio della macchina/impianto (visualizzazione della tabella dei dati definita + eventuali ulteriori funzioni)
3	Test del collegamento PC ↔ PLC Siemens S7-1200 in ufficio (rete locale)
4	Test del collegamento PC ↔ PLC Siemens S7-1200 in ufficio con ponte hotspot cellulare (da remoto)
5	Invio notifica e-mail in caso di valori anomali o di allarmi sulla macchina
6	Test di connessione con 2 PLC Siemens S7-1200 attivi
7	Test di connessione a PLC Siemens S7-1200 via dispositivo di connessione remota (Secomea)
8	Test di connessione a PLC Siemens S7-1200 via VPN
9	Implementazione su PLC Siemens S7-1200 di un sistema di invio notifica alla Messersì per presenza allarmi o necessità di manutenzione
10	Test del sistema con PLC di brand diversi da Siemens

Tabella 2.1 - Fasi progettuali

2.3 Specifiche implementate

Come definito, il primo passo è stato definire le tabelle per lo scambio dati con il supervisore. Nello specifico, sono state previste due DB dati da inserire nei PLC che dovranno connettersi a Messersì Supervisor.

La prima, chiamata DB_Receive (tabella 2.2), sarà utilizzata per elaborare i comandi ricevuti dalla app, quali reset e settaggio del conteggio dei cicli di produzione e impostazione dello stato di funzionamento della macchina.

SUPERVISIONE → MACCHINA (DB100) - 32 byte			
INDIRIZZO	**DATO**	**FORMATO**	**DESCRIZIONE**
0.0	**Watchdog**	BOOL	Clock 1s per il controllo di comunicazione attiva.
0.1	**Reset conteggio cicli eseguiti**	BOOL	
0.2 ÷ 3.7	**Scorta**	BOOL	
4.0	**Stato macchina**	INT	Stato macchina: 0: Emergenza 1: Manuale 2: Automatico 3: Anomalia 4: Esclusione
8.0	**Numero cicli eseguiti**	DINT	N° di cicli eseguiti dopo l'azzeramento del contatore.
4.0 ÷ 31.7	**Scorta**	INT	

Tabella 2.2 - Scambio dati App -> Macchina

La DB_Send (tabella 2.3) si occuperà invece di inviare i dati da mostrare in app.

MACCHINA → SUPERVISIONE (DB101) - 64 byte			
INDIRIZZO	**DATO**	**FORMATO**	**DESCRIZIONE**
0.0	**Watchdog**	BOOL	Clock 1s per il controllo di comunicazione attiva.
0.1 ÷ 1.7	**Scorta**	BOOL	
2.0-3.7	**TABELLA 16 ALLARMI PRINCIPALI**	BOOL	
6.0	**Stato macchina**	INT	Stato macchina: 0: Emergenza 1: Manuale 2: Automatico 3: Anomalia 4: Esclusione
8.0	**Numero cicli eseguiti dall'ultimo reset**	DINT	N° di cicli eseguiti dopo l'azzeramento del contatore.
12.0	**Numero totale cicli eseguiti**	DINT	N° di cicli eseguiti dall'avviamento della macchina.
16.0	**Numero ore di lavoro dall'ultimo reset**	DINT	N° di ore di lavoro dall'ultimo reset
20.0	**Numero totale ore di lavoro**	DINT	N° di ore di lavoro dall'avviamento della macchina.

Tabella 2.3 - Scambio dati Macchina -> App

Infine, tramite clock di frequenza 1 Hz vengono scambiati segnali watchdog per verificare la corretta comunicazione tra le parti.
Inizialmente è stato previsto di realizzare un'applicazione desktop contenente quattro pagine principali:

1. Ricerca nuova macchina
2. Check connessioni attive
3. Comando macchina
4. Monitoraggio segnali da PLC

A queste si sono poi aggiunte una interfaccia di login e una di impostazioni. La prima consente l'accesso al supervisore solo al personale autorizzato, mentre la seconda è utilizzata per visualizzare e modificare dati riguardanti le macchine a cui è possibile connettersi e utenti autorizzati.
Queste informazioni sono archiviate in un database, posto nella rete aziendale, a cui è necessario connettersi per poter utilizzare l'app.
È stato anche implementato il sistema notifica e-mail in caso di rilevazione di uno o più allarmi attivi.
Sono stati previsti due casi d'uso dell'applicazione: il primo ne presume l'utilizzo da parte dell'assistenza clienti della sede di Barbara. Il secondo presuppone anche l'uso sul posto, in officina in fase di collaudo o presso il cliente dopo l'installazione. In questa ottica è stato considerato opportuno sviluppare una versione mobile per smartphone e tablet Android. Le uniche differenze rispetto alla versione desktop sono l'assenza del sistema di notifica allarmi e lo scambio di watchdog.
È possibile connettersi solamente a macchine registrate nel database, in modo da garantire la connessione solo a indirizzi verificati. I PLC supportati sono quella della gamma Siemens S7, ovvero i modelli i PLC S7-200, Logo 0BA8, S7-300, S7-400, S7-1200 e S7-1500.
Le informazioni di registro di ciascuna macchina connessa vengono mostrate nella pagina delle connessioni attive. È possibile visualizzare il nome assegnato al PLC e matricola, modello e cliente riguardanti la macchina. Inoltre, ad ogni modello è assegnato un campo "gruppo" per identificare il tipo di macchinario (reggiatrice orizzontale, verticale, incappucciatrici, ecc.).
Il controllo è invece un'istruzione da utilizzare con cautela e, secondo le norme di sicurezza, solo sul luogo. È quindi una funzionalità da usare preferibilmente dalla app mobile. Sarà necessario modificare i programmi PLC nel caso si vogliano implementare particolari controlli, quali la movimentazione di parti della macchina o l'avvio del ciclo di produzione.
Nella pagina di monitoraggio sono visibili le tabelle di scambio dati, si può controllare se sono attivi allarmi e verificare lo stato di ingressi e uscite del PLC.

2.4 Connessioni

L'applicativo è in grado di mantenere connessioni TCP/IP con uno o più PLC connessi in locale o da remoto. Inoltre, più dispositivi possono usare l'app Messersì Supervisor per connettersi allo stesso PLC nello stesso momento.

Per il funzionamento del sistema gli impianti dovranno essere predisposti al collegamento con Messersì Supervisor e sarà necessario configurare i PLC per la connessione ad Internet e quindi all'applicazione.

Il collegamento potrà avvenire tramite IP locale o remoto: nella seconda ipotesi si possono utilizzare appositi gateway industriali per la connessione ad Internet dei dispositivi dell'impianto quali i PLC.

L'applicazione va poi a collegarsi con la rete aziendale Messersì, dove è stata attivata una macchina virtuale che funge da host per il database contenente i dati necessari al funzionamento del sistema.

Si può quindi monitorare la produzione sia sul luogo che ovunque tramite Internet, su computer e dispositivi mobile Android.

Figura 2.2 - Connessioni nel sistema Messersì Supervisor

Verranno illustrati in seguito gli strumenti e le tecnologie coinvolte nella realizzazione di questo progetto, entrando nel dettaglio delle funzioni e degli obiettivi raggiunti.

Capitolo 3
Internet of Things e Industria 4.0

"Siamo in continua ricerca, perché chi fa ricerca si pone avanti mentre gli altri restano indietro a copiare. Oggi, per sostenere questo progetto e questa azienda, abbiamo bisogno di gente che ci conosca e che sia motivata"[2]

3.1 La quarta rivoluzione

Così parlava Maurizio Messersì il 19 ottobre 2016, durante la conferenza indetta per la stipulazione dell'accordo tra la Messersì Packaging e l'Università Politecnica delle Marche, un progetto di durata triennale che prevedeva l'inserimento di studenti dell'ateneo nell'azienda a fronte di un arricchimento del know-how produttivo in termini di innovazione, sostenibilità ed efficienza dell'azienda e degli studenti stessi.

Al contempo, anche l'ex rettore Sauro Longhi indicava questa via come un primo esempio di come l'innovazione stava entrando nei processi manifatturieri, in quello che veniva indicato come Industria 4.0.

L'impatto delle nuove innovazioni si inserisce in un contesto più ampio, quello della quarta rivoluzione industriale. Per ogni rivoluzione, è identificabile un momento che segna il punto di non ritorno, in cui il cambiamento va imponendosi inarrestabilmente.

Questo vale anche per le rivoluzioni industriali: la prima ebbe luogo in Inghilterra a fine '700 con l'invenzione della macchina a vapore e l'introduzione delle macchine per mansioni che fino ad allora erano state eseguite da operatori manuali o animali.

Alla fine del XIX secolo fu l'energia elettrica a segnare la seconda rivoluzione industriale con l'introduzione della catena di montaggio e la nascita del sistema capitalistico industriale, mentre nella seconda metà del '900 si è assistito alla terza rivoluzione, definita "rivoluzione digitale", segnata dallo sviluppo dei semiconduttori e quindi dalla nascita del computer.

Le nuove tecnologie protagoniste di questa rivoluzione sfrutteranno invece i mezzi offerti dalla digitalizzazione e dall'informatica, tra tutti Internet.

Secondo il professor Klaus Schwab, fondatore e presidente del World Economic Forum, sono tre i parametri caratterizzanti di questa quarta rivoluzione:

[2] https://messersi.com/it/news-eventi/la-ricerca-disposizione-dellindustria-40/

- La velocità con cui si sta diffondendo.
- La portata del cambiamento e i cambi di paradigma sia nell'individuo che nella società.
- L'impatto che avrà su sistemi, paesi, aziende e l'intera società.

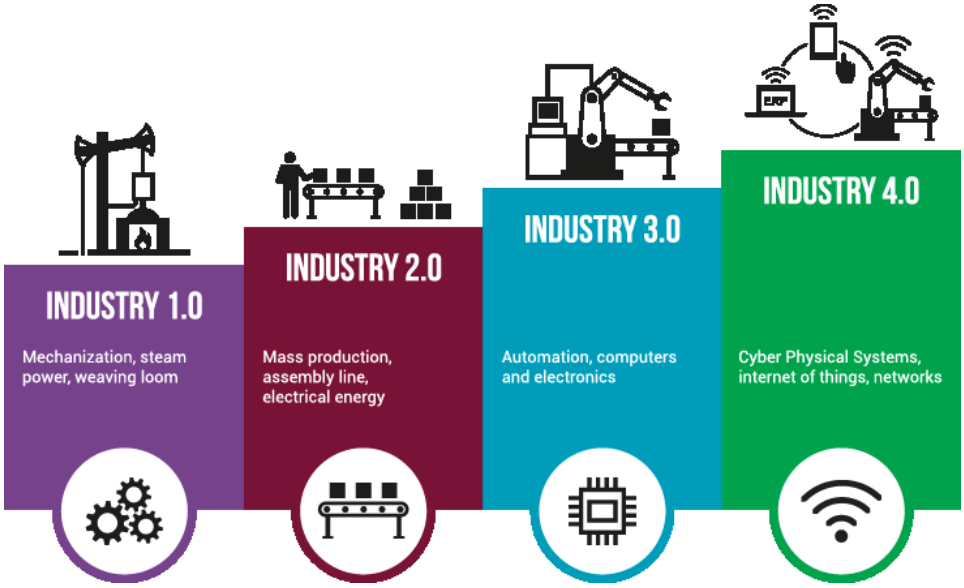

Figura 3.1 - Highlights of Industrial Revolutions, reseatchgate.net

3.2 Internet of Things

Uno dei principali sviluppi di questa rivoluzione è l'Internet of Things (IoT, letteralmente "internet delle cose"), termine con cui si definisce l'insieme delle tecnologie atte a mettere in comunicazione dispositivi più o meno complessi attraverso Internet.

Si tratta di un termine coniato nel settore delle telecomunicazioni, coniato per la prima volta da Kevin Ashton, ricercatore presso il MIT, Massachussets Institute of Technology, dall'esigenza di dare un nome agli oggetti connessi ad internet.

Tramite questa rete, formata da sensori e dispositivi di qualsiasi livello di intelligenza, si viene a creare un continuo scambio di dati tramite la connessione ad Internet.

I dispositivi che sapranno approfittare di quest'aumento dell'informazione e saranno in grado di processarla al meglio grazie alla loro intelligenza, potranno operare con performance maggiori e implementare nuove funzioni.

3.2.1 Architettura di un sistema IoT

Attualmente non esiste uno standard unico per la realizzazione di sistemi che sfruttino tecnologie IoT, seppur esistano vari modelli adeguabili alle diverse esigenze.

La raccolta dell'informazione (*data collection)* è affidata a sensori e unità basate su microcontrollore (MCU) o a microprocessore (MPU) più o meno complesse, le quali si occupano anche della loro processazione.
Poiché le unità devono comunicare tramite Internet, si rende necessario l'impiego di gateway, dispositivi di rete che connettono le unità IoT al cloud, formando una mappa intelligente delle "cose", in continuo scambio di conoscenza.
Ovviamente, non è sufficiente ricevere un segnale fisico per rendere l'informazione utilizzabile dall'utente. È necessaria un'accurata scelta delle tecnologie, a partire dai mezzi fisici per la comunicazione, fino ai software con cui l'utente si interfaccerà.
Per gestire l'informazione tramite Internet ci si adegua al modello OSI, che definisce l'architettura della logica di rete nelle telecomunicazioni e nell'informatica.
Il modello è organizzato gerarchicamente in sette livelli, con diverse funzionalità a livello hardware e software. Tralasciando la descrizione dei vari livelli, tratteremo le tecnologie e i protocolli impiegati nello sviluppo della tesi.

Figura 3.2 - Modello OSI

Al primo livello vengono messi a disposizione protocolli per la definizione delle caratteristiche fisiche del mezzo di trasmissione. La soluzione migliore dipende da fattori quali distanza da percorrere, quantità di dati trasmessa e latenza ammessa.
Di seguito gli standard impiegati a livello fisico per la connettività della app Messersì:

- Ethernet: connessione cablata poco costosa veloce e a bassa latenza.
- Reti mobili: sfruttano i pacchetti dati cellullari come trasporto dati. Al momento, le reti mobili a maggiore capacità e velocità sono quelle 4G LTE (fino a 100 Mbps) e 5G, attualmente in diffusione, capace di raggiungere velocità di qualche Gigabit per secondo.

A causa delle sue alte frequenze, la rete 5G necessita di una maggiore infrastruttura (antenne e ripetitor). La diffusione della rete 5G, seppur fonte di controversie, sarà indispensabile per la realizzazione di una società delle cose.

- Wi-Fi 802/11: protocollo wireless a basso costo di gestione, ideale per scenari domestici o di ufficio. Tuttavia, non dispone di una notevole portata, e necessita di essere alimentato costantemente.

Altri protocolli particolarmente noti sono il Bluetooth, NFC e RFID, utilizzate per trasmissioni dati a corto raggio.

Nella comunicazione Internet si impiega invece il protocollo di rete IP (terzo livello OSI), nelle varianti IPv4 o IPv6. Il protocollo IP è responsabile del corretto instradamento dei dati, consentendo la comunicazione tra diverse reti fisiche.

I dati vengono racchiusi in un pacchetto a 32 bit, costituito da un header contenente le sue proprietà strutturali e seguito dall'informazione vera e propria.

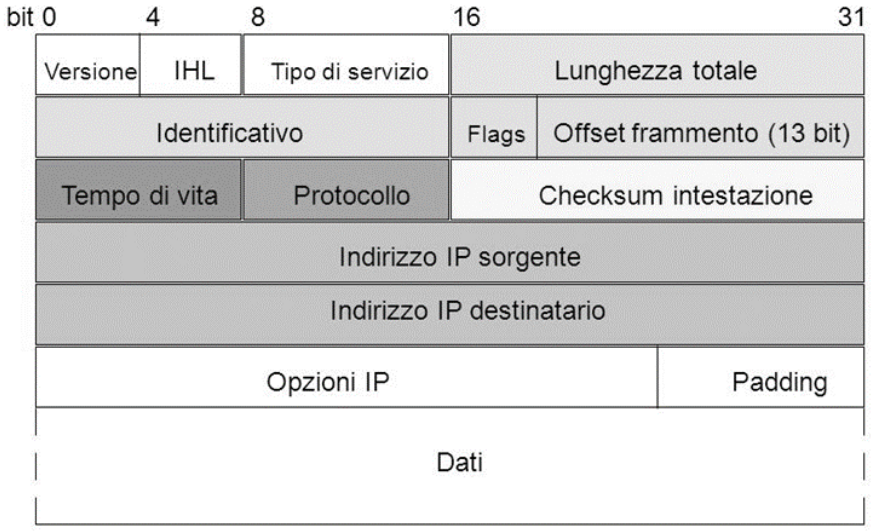

Figura 3.3 - Pacchetto dati IP

Attraverso un indirizzo univoco a 32 bit, ciascun dispositivo viene identificato all'interno della rete, permettendone la comunicazione e lo scambio dati.

Inoltre, si necessita un canale di comunicazione end-to-end che protegga i dati durante il trasferimento tra i livelli. Di ciò ci si occupa a livello trasporto, con il protocollo TCP (Transmission Control Protocol), che fornisce la verifica della corretta consegna del pacchetto da host a host.

Per quanto riguarda il livello applicazione, il più alto nella scala gerarchica, sono diffusi vari protocolli a seconda degli ambiti applicativi.

3.2.2 Impieghi

Nel 2015 il World Economic Forum ha stilato un elenco di 21 "punti di discontinuità", ovvero 21 innovazioni in tutti i settori, che, secondo più di 800 esperti, entro il 2025 segneranno il passaggio verso il mondo digitalizzato e interconnesso.

Di seguito analizzeremo due esempi per rendere l'idea della forza dell'innovazione portata dall'IoT in tutti i settori.

3.2.2.1 Smart mobility

Uno dei 21 punti prevede che entro il 2025 si giunga alla prima città con più di 50 mila abitanti priva di semafori.

Città, infrastrutture e strade potranno adeguarsi per usufruire della quantità di informazioni messe a disposizione dall'Internet of Things per gestire in maniera più intelligente ed efficiente i flussi di traffico.

La svolta in tale direzione è già avvenuta e le società si stanno già adeguando al cambiamento: BMW stima che presto il 22 % dei veicoli (290 milioni) sarà connesso a Internet[3] e aziende automobilistiche e del digitale come Audi e Google stanno unendo gli sforzi per far sì che questa fusione di idee entri in funzione il prima possibile.

Città all'avanguardia quali Singapore e Barcellona stanno già mettendo a punto sistemi di questo tipo per ridurre code e affollamenti. Ad esempio, nella città catalana sistemi di sensori sono stati impiantati nel mando stradale per rilevare la presenza di parcheggi liberi, mentre il modello *Superblock* ha riorganizzato la rete dei trasporti analizzando la topografia dei quartieri, in modo da smaltire il traffico e fornire nuove linee metropolitane o parcheggi sotterranei per liberare spazio in superficie[4].

La quantità di informazioni messa a disposizione dall'IoT potenziano quindi le opportunità del *data analytics,* in modo da migliorare il modo di vivere la città.

3.2.2.2 Agricoltura 4.0

La capacità di comunicare con l'esterno il proprio stato, la raccolta di dati grezzi e da essi il patrimonio informativo ricavato, rendono possibile sbloccare nuovi orizzonti tecnologici in tutti i settori produttivi.

A dimostrazione di tale versatilità, possiamo prendere ad esempio ciò che sta comportando l'avvento di queste tecnologie in un ambito proveniente da un contesto totalmente opposto al digitale: l'agricoltura.

[3] https://www.politico.eu/article/google-vs-german-car-engineer-industry-american-competition/
[4] https://www.babilonmagazine.it/barcellona-nuovo-paradigma-per-la-smart-city/

Quella che indicata con Agricoltura 4.0. sfrutta sensori installati nel terreno per estrapolare dati su irraggiamento, temperature, stress idrico e molto altro, da utilizzare poi durante i processi di produzione, che spesso necessitano di continua attenzione e di strumentazioni di precisione all'avanguardia. Le tecnologie applicate vanno dal Data Analytics alle mobile app, passando per l'impiego di mezzi autonomi (droni e robot) e fino all'intelligenza artificiale, come testimonia l'indagine svolta dall'Osservatorio Smart AgriFood su 218 startup italiane del settore[5]:

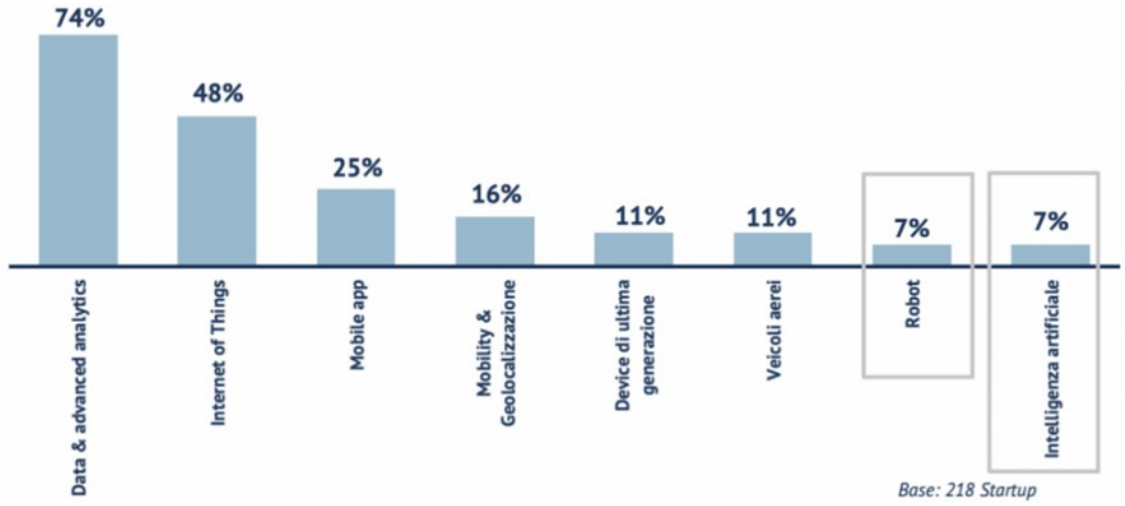

Figura 3.4 – Start Up e Smart Agrifood: le tecnologie adottate, agrifood.tech

3.3 Industria 4.0

L'implementazione di sistemi e applicazioni smart nell'ambito dell'automazione industriale è presente da anni nelle linee di produzione, tanto che nel nostro paese comportava un business di circa un miliardo e 200 milioni di euro già nel 2015[6].

Nel contesto della quarta rivoluzione industriale, anche il settore manufatturiero ha raggiunto notevoli progressi in termini di automazione dei processi e interconnessione degli strumenti di produzione. Questo percorso di evoluzione è indicato con il termine Industria 4.0.

Industria 4.0 è un nuovo modo di concepire il settore, attraverso l'impiego di alcune tecnologie, dette abilitanti, tra le quali robot collaborativi, stampanti 3D, realtà aumentata e *data analytics*.

[5] https://www.agrifood.tech/internet-of-things/lagricoltura-4-0-va-vesro-linternet-of-things/
[6] https://www.internet4things.it

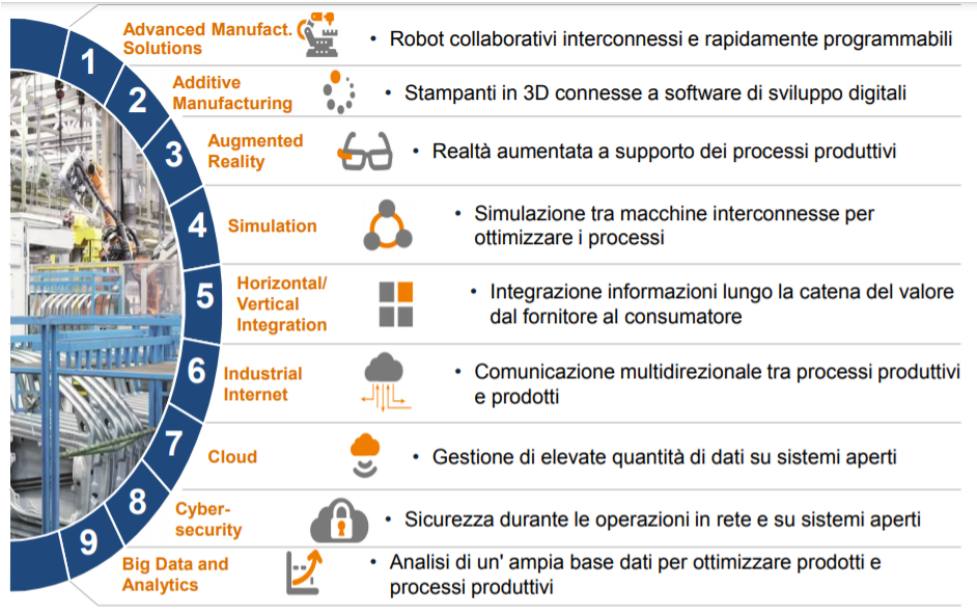

Figura 3.5 - Tecnologie abilitanti

Alcune di queste tecnologie sono derivate dall'Internet of Things e sono necessarie per la realizzazione di sistemi rientranti nei canoni di Industria 4.0.

L'Industrial Internet e quindi la *smart factory* sono uno dei mattoni di Industria 4.0, in quanto viene permessa la comunicazione tra processi produttivi e sistemi fisici e digitali, rendendo le macchine interconnesse tramite Internet.

Grazie all'enorme quantità di dati proveniente da reti costituita da macchine, sensori, applicativi specifici e molto altro, è possibile ottenere informazioni riguardo lo stato della rete produttiva e di conseguenza migliorare l'intera catena di lavorazione.

Il primo requisito per la conversione verso lo smart manufacturing è la riorganizzazione dei processi e la raccolta dati tramite strumenti di analytics che consentano la raccolta e la gestione delle informazioni produttive.

È poi indispensabile dotare macchine di sensori e dispositivi smart in grado di connettersi ad Internet per comunicare alla rete ed eventualmente al cloud informazioni sullo stato ambientale e di esercizio della linea.

Sistemi gestionali, piattaforme online e mobile app permettono poi l'interazione macchina-azienda e azienda-cliente. In questo modo si ha il controllo totale sullo stato dell'impianto, consentendo di identificarne punti di forza e criticità.

L'automazione spinta e le soluzioni IT permetteranno agli impianti di evolversi verso una dimensione interconnessa, controllabile anche da remoto tramite strumenti di monitoraggio.

Proprio in questa direzione è stato portato avanti il progetto svolto con Messersì Packaging: realizzare un sistema interconnesso con la linea di produzione in modo da migliorare produttività, assistenza al cliente e prevenire guasti o malfunzionamenti.

3.3.2 Piano nazionale Industria 4.0

Per i paesi occidentali, le nuove tecnologie costituiranno una sfida di notevole importanza: la digitalizzazione può fornire gli strumenti per recuperare il vantaggio competitivo che le industrie dei paesi più sviluppati sembrano aver perso a causa della globalizzazione. L'importanza di investire in questo ambito è rimarcata anche dal piano di incentivi varato dal Governo italiano.
Da sempre per l'Italia il settore industriale ha rappresentato il motore della crescita e dello sviluppo economico, di ricchezza e occupazione. Con la Legge di Bilancio 2017 il Governo ha dato attuazione al Piano Nazionale Industria 4.0 che ha riportato la politica industriale al centro dell'agenda legislativa.
"Quello che il Governo propone è un patto di fiducia con il mondo delle imprese che vogliono crescere e innovare" – Carlo Calenda, Ministro dello Sviluppo Economico dal 2016 al 2018
Il piano si basa su tre pilastri:

- Incentivi all'investimento nell'Internet delle cose, con ammortamenti fino al 150 % (Legge Sabatini).
- Rivalutazione delle università come centri di formazione sull'IoT, che dovranno essere sempre più vicine all'impresa nel percorso di innovazione.
- Stesura di uno standard per l'adozione di tecnologie IoT, ma lasciando che il piano sia "aperto" a tutti, senza vincoli su piattaforme, infrastrutture e ambienti operativi.

Il Piano Nazionale Industria 4.0 offre alle aziende italiane strumenti per cogliere le opportunità dell'innovazione e del digitale legate alla quarta rivoluzione industriale, tramite bonus e ammortamenti per l'acquisto di macchinari industriali dotate di sistemi di telemanutenzione, telediagnosi, controllo in remoto, monitoraggio continuo delle condizioni di lavoro e dei parametri di processo. Inoltre, vengono fortemente incentivate, ricerca e formazione di personale nell'ottica di tale riforma.
Ancora oggi, nonostante l'attuale situazione di crisi dovuta all'epidemia Covid-19, Industria 4.0 resta al centro dell'agenda di Governo, tant'è che tramite i fondi del Recovery Fund, il fondo europeo costituito per sostenere la ripresa delle economie dell'Unione, il nuovo piano nazionale sarà rinnovato[7].

[7] https://www.ansa.it/

3.4 Impatto della rivoluzione

L'enorme portata di innovazioni e cambiamenti innescati dalla quarta rivoluzione sta portando cambiamenti sia sull'individuo che nel collettivo, per le cui conseguenze sono al momento impossibili da prevedere.

Lo scopo di questo capitolo è quello di fornire alcuni esempi di tali innovazioni sulla base delle opinioni degli esperti, in modo da descriverne le principali criticità e i possibili miglioramenti sulle nostre vite, sia in ambito economico che sociale.

3.4.1 Riorganizzazione della forza lavoro

È opinione diffusa tra gli esperti che la quarta rivoluzione industriale avrà effetti talmente vasti ed eterogenei sull'economia mondiale che sarà difficile classificarli in maniera precisa.

Il primo dilemma riguarderà l'impatto di questa rivoluzione sui livelli di occupazione. Già nel 1931 l'economista John Keynes prevedeva una vasta disoccupazione a causa delle scoperte che avrebbero permesso di aumentare i ritmi produttivi sostituendo il lavoro umano.

Per comprendere queste preoccupazioni, basti pensare che nel 2014, le tre maggiori aziende della Silicon Valley hanno ottenuto una capitalizzazione di 1,09 trilioni di dollari con un numero di dipendenti pari 137 mila lavoratori. Nel 1990, le tre maggiori aziende di Detroit avevano invece avuto una capitalizzazione di 36 miliardi, impiegando ben 1,2 milioni di lavoratori[8].

Si andrà quindi verso una nuova organizzazione dei reparti produttivi, specialmente nell'ambito manufatturiero, dove verosimilmente la manodopera verrà sempre più sostituita da macchinari. Allo stesso tempo si potrebbe però assistere a un aumento dell'occupazione per altri impieghi, specialmente professioni intellettuali, creative o che necessiteranno di competenze sociali e decisionali.

3.4.2 Internet sempre e ovunque

In secondo luogo, vivremo in un mondo in cui saremo perennemente connessi e saremo circondati da dispositivi che modificheranno sempre più le nostre abitudini.

Le politiche di innovazione tecnologica mirano sempre più a creare dispositivi miniaturizzati e con maggiori capacità. Ne consegue una riduzione di prezzo che rende questi dispostivi sempre più diffusi, tanto che oggi quasi sei miliardi di persone hanno accesso a uno smartphone[9].

Conseguenza diretta sarà la diffusione di Internet in tutto il globo, che consentirà l'accesso all'informazione anche alle popolazioni finora più arretrate.

[8] James Manyika, Michael Chui, *Digital Era Brings Hyperscale Challenges"*, The Financial Times, 2014
[9] https://tg24.sky.it/tecnologia/2020/03/04/smartphone-mondo

Se nei paesi già questo potrebbe solamente comportare un miglioramento nella soddisfazione del cliente, nei paesi più poveri potrebbe significare dare accesso ai servizi fondamentali, quali scuola e sanità, a quella parte di popolazione più svantaggiata che finora ne è rimasta esclusa.

Il progetto sviluppato ha quindi l'intento di trattare alcuni temi di Industria 4.0: partendo dal gran numero di sensori e attuatori impiegati in una linea per l'imballaggio, sfrutteremo i PLC, dispositivi di elaborazione di carattere industriale, per creare l'infrastruttura intelligente tramite la quale l'applicazione permetterà lo scambio in tempo reale delle informazioni ricavate dai sistemi produttivi.
In questo modo, le installazioni sparse in ogni angolo del mondo potranno essere monitorate dalla sede centrale di Barbara o da ovunque ci si trovi.

Capitolo 4
Il PLC

È indispensabile definire il ruolo dell'unita di elaborazione nei sistemi oggetti del controllo di questa applicazione: il PLC. Oltre ad una sua descrizione, vengono illustrate le tecnologie impiegate nei sistemi automatizzati Messersì Packaging.

4.1 Cenni sull'automazione industriale

Per poter realizzare un sistema di supervisione valido, è necessario conoscere l'architettura e il funzionamento delle linee produttive.

Nell'industria attuale l'automazione ha ormai un ruolo dominante, specialmente nello svolgimento di operazioni ripetitive o di alta precisione, consentendo di ottimizzare i processi sia in termini di velocità che di qualità del prodotto finale.

In un sistema automatizzato si elaborano input provenienti da una parte controllata, per poi eseguire azioni coordinate con gli elementi del sistema, opportunamente programmati e impostati. Si possono allora raggiungere diversi livelli di complessità dell'automazione, per una o più macchine, ma con orizzonti ampliabili fino ad una completa automatizzazione della fabbrica, dove l'uomo ha il solo compito di supervisore del processo.

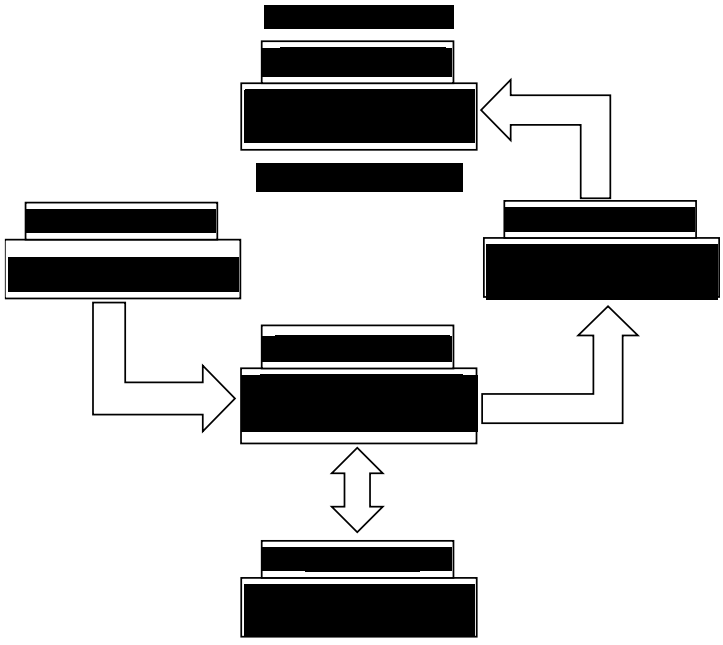

Figura 4.1 - Componenti di un sistema automatizzato

Ogni sistema è organizzato in una parte *operativa* comprendente le sue componenti azionatrici quali motori, cilindri, ecc. e in una parte di *comando,* la quale racchiude l'unità di controllo, ovvero il dispositivo di elaborazione dell'informazione (PLC, microcontrollori) e i sensori tramite i quali vengono rilevati gli stati del processo.

4.1.1 Tecnologie impiegate

Oltre al dispositivo di elaborazione, un sistema automatizzato necessita di componenti in grado di rilevare l'informazione dall'ambiente e di dispositivi in grado di operare azionamenti sulla base dei dati a loro forniti.

Sono perciò necessari dispositivi per la lettura delle condizioni rilevanti e il conseguente invio dati all'unità centrale. Si possono rilevare svariate informazioni, quali movimenti, velocità o posizionamenti e si può acquisire l'informazione su praticamente qualsiasi grandezza fisica. A seconda dell'impiego si utilizzano sensori digitali o analogici che possono essere di tipo meccanico (fine corsa, encoder), capacitivi, induttivi o fotocellule.

Una volta elaborata l'informazione, è necessario pilotare i dispositivi attuatori sulla base delle istruzioni del programma. Tramite driver di potenza vengono pilotati azionamenti di tipo elettromeccanico (motori in corrente continua, alternato, passo-passo, ecc.) o pneumatico (cilindri, valvole, ecc.).

Infine, spesso si fornisce un'interfaccia di comunicazione uomo-macchina per permettere all'operatore di intervenire nella gestione del sistema tramite menù grafici di configurazione. Queste interfacce possono svariare da semplici pulsantiere a terminali grafici complessi come display touchscreen.

Figura 4.2 - Esempio di linea automatizzata Messersì, Messersi.com

4.2 Architettura

Il PLC, *Programmable Logic Controller,* nasce negli anni ’70 come sistema di controllo per apparecchiature pesanti, andando a sostituire i tradizionali sistemi a logica cablata e relè.

Per definizione, il PLC è un *sistema elettronico digitale destinato all’uso industriale, che utilizza una memoria programmabile per l’archiviazione interna di istruzioni, orientata all’utilizzatore per l’implementazione di funzioni specifiche (logiche, di sequenziamento, di temporizzazione, conteggio, ecc.) e per controllare, mediante ingressi e uscite digitali e analogici, vari tipi di macchine.*[10]

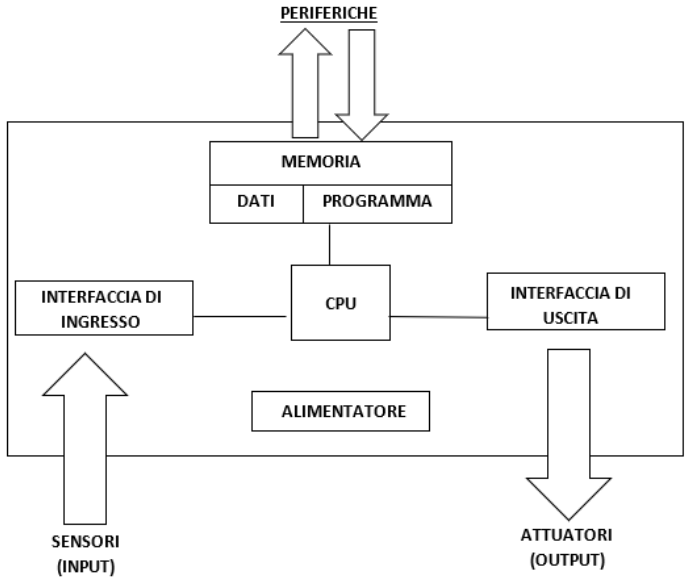

Figura 4.3 – Schema a blocchi di un PLC

Nell’ambito della fabbrica automatizzata i PLC si pongono da intermediatori tra elettromeccanica ed elettronica. Permettono infatti l’esecuzione di processi più o meno complessi tramite la scrittura di codice da parte del programmatore.

A questo proposito l’International Electrotechnical Commission fornisce uno standard di linguaggi per controller programmabili.[11] Per quanto riguarda i PLC, vengono definiti cinque linguaggi di programmazione, tre di tipo grafico e due testuali. Nella prima categoria rientrano i ladder diagram (LD), sequential function block (SFC) e function block diagram (FBD), mentre nella seconda si trovano l’instruction list (IL) e lo structured test (ST). Tramite un compilatore il programma viene poi tradotto in linguaggio macchina per l’esecuzione da parte della CPU.

Nonostante la minore potenza di elaborazione, nell’ambito industriale i PLC si fanno preferire ai microcontrollori per la loro facilità di gestione, programmazione, flessibilità e, non

[10] Norma IEC 1131
[11] Norma IEC 61131

secondariamente, per la loro adeguatezza ad operare in ambienti difficili e con presenza di disturbi elettrici e urti meccanici, tipici degli ambienti industriali.
Il funzionamento di un PLC è caratterizzato da cicli in cui si susseguono tre fasi:

1. Acquisizione dei dati in ingresso da parte della memoria interna
2. Esecuzione sequenziale delle istruzioni del programma
3. Aggiornamento dello stato delle uscite

Esistono varie tipologie di PLC, classificabili a seconda di dimensione e numero di ingressi e uscite. Si possono perciò avere PLC compatti, modulari ed espandibili. Tutti però condividono la stessa architettura interna, tramite la quale la CPU comunica con il mondo esterno.
I PLC espandibili godono della possibilità di ampliamento tramite moduli che possano aumentare il numero di ingressi e uscite, implementare funzioni caratteristiche l'interfacciamento con dispositivi quali termocoppie o moduli per la comunicazione.
Infine, anche i software di sviluppo mettono a disposizione funzioni avanzate per impieghi di alto livello. Sempre più applicazioni di automazione sfruttano blocchi software che implementano regolatori PID, generatori di segnali PWM e azionamenti SERVO.

4.3 Siemens S7

L'offerta del mercato dei PLC è ormai ampia, per un settore di cui fanno parte tutti i maggiori brand del settore dell'automazione. Al giorno d'oggi, le differenze di funzionamento tra una casa costruttrice e l'altra sono però pressoché trascurabili.
Tuttavia, per lo sviluppo di Messersì Supervisor, il campo è stato ristretto ai dispositivi di marca Siemens, per alcune ragioni:

1. Sono i più impiegati nelle soluzioni Messersì.
2. È disponibile una libreria open source basata su framework .NET che offre funzioni per il collegamento tramite istruzioni in linguaggio C#.
3. Offrono un ambiente di sviluppo intuitivo che facilità la programmazione e il controllo *real time* in caso di necessità.

Sempre per la maggiore diffusione nell'ambito Messersì, analizzeremo le principali caratteristiche del modello S7-1200.
Si tratta di un dispositivo modulare compatto, utilizzato in genere per sistemi di automazione di media fascia ad alte prestazioni e ad alta versatilità.
Durante lo sviluppo è stata utilizzata la versione con a bordo la CPU denominata "1214C DC/DC/DC".

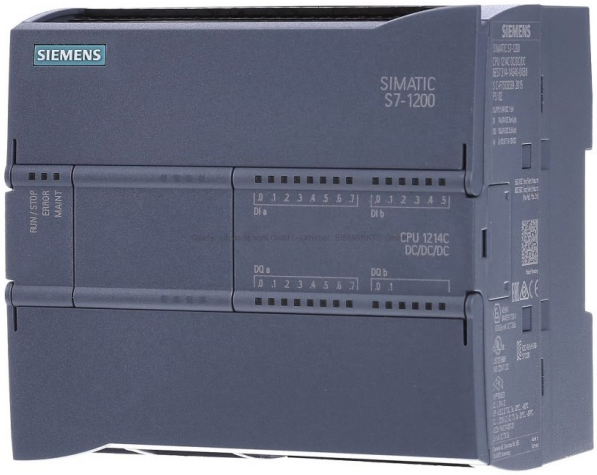

Figura 4.4 - PLC S7-1200

In tabella 4.1 vengono ricapitolate le principali caratteristiche tecniche del dispositivo:

Memoria di lavoro	125 Kb
Memoria di caricamento	4 Mb
Ingressi digitali	14
Ingressi digitali veloci	6 (dei 14 ingressi digitali)
Ingressi analogici	2 (0 ÷ 10 Vcc)
Uscite digitali	10
Uscite PWM	4 (delle 10 uscite digitali)
Alimentazione	24 Vcc

Tabella 4.1 - Caratteristiche del PLC Siemens S7-1200

Il PLC S7-1200 dispone inoltre di porta Ethernet industriale per la programmazione e la connessione a interfacce esterne, quali dispositivi HMI o reti Profinet.
L'area di memoria dei PLC Simatic è strutturata in quattro diverse aree, ciascuna con indirizzi univoci:

- Memoria globale: area specializzata per la memorizzazione di ingressi, uscite e variabile interne (*merker*), accessibile da tutti i blocchi del codice.
- Tabella delle variabili: consente di assegnare nomi simbolici per indirizzi di memoria specifici, richiamabili in tutto il programma.
- Blocco dati (DB): globali, cioè accessibili da ogni parte del codice, o di istanza, cioè usufruibili solo da alcuni blocchi. Utilizzati per mantenere dati memorizzati in memoria anche dopo l'esecuzione del codice associato.

- Memoria temporanea: ad alta velocità, riceve dalla CPU il blocco di codice da eseguire ad ogni call

Per eseguire le istruzioni desiderate, sono impiegabili tre tipi di blocchi in cui scrivere il codice utente:

- Blocchi organizzativi (OB): definiscono la struttura del programma e fungono da interfaccia tra il sistema operativo e il programma utente. Vengono infatti eseguiti dalla CPU quando si verifica uno specifico evento (allarmi, intervalli di tempo, ecc.). Ne è un esempio l'OB di ciclo, eseguito dal programma ad ogni ciclo CPU.
- Funzioni (FC): blocchi di codice che eseguono operazioni specifiche a partire da valori di ingresso. Sono richiamabili anche più volte e da qualsiasi punto del programma. Salvano i risultati delle operazioni in locazioni di memoria temporanee.
- Blocchi funzionali (FB): sottoprogrammi la cui esecuzione deve essere richiamata da un altro blocco (OB, FC o FB). Memorizzano i propri dati in modo permanente, in modo da poterne disporre anche dopo l'elaborazione del blocco.

Le possibilità di personalizzazione del codice sono innumerevoli anche a seconda dell'ambiente di sviluppo, il TIA Portal nel caso Siemens.

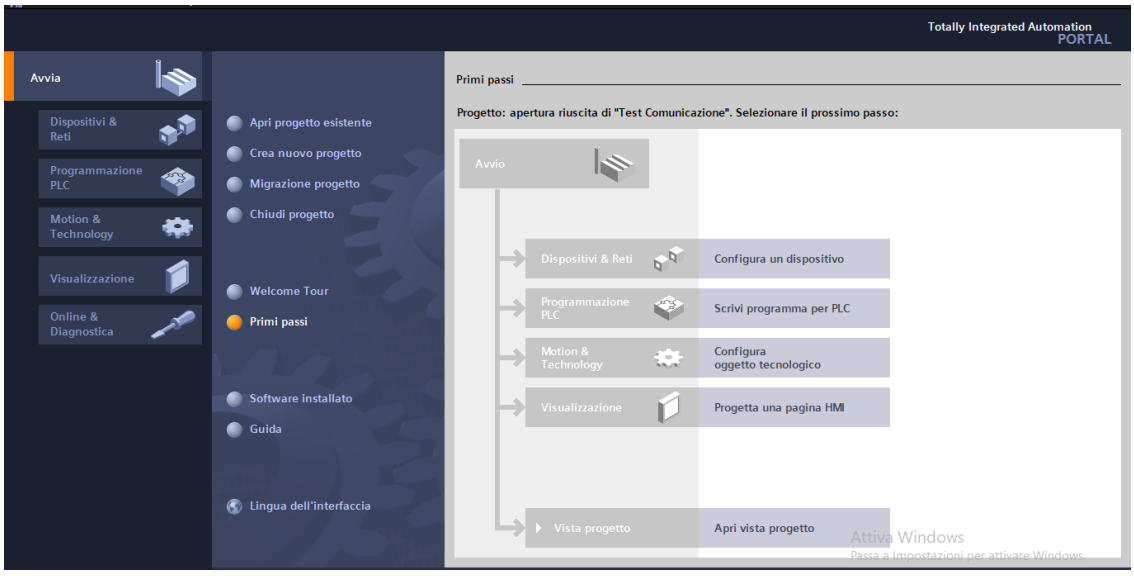

Figura 4.5 - Schermata di avvio TIA Portal

In TIA Portal, i linguaggi di programmazione utilizzabili sono il Ladder e il linguaggio KOP, anch'esso di tipo grafico.

Vengono messe a disposizione cinque classi di operazioni:

- Istruzioni di base: operazioni logiche e aritmetiche, timer, contatori e funzioni matematiche.

- Istruzioni avanzate: stringhe, allarmi, diagnostica, controlli PID, motion control, ecc.
- Di comunicazione: protocollo S7, web servers, ecc.
- Pacchetti opzionali.

L'insieme dei blocchi contenente il codice utente forma il programma che controlla il processo produttivo. È quindi essenziale prevedere tutti i possibili stati durante l'esecuzione, prevenire guasti e situazioni indesiderate e gestire i malfunzionamenti.

4.4 Modellazione di un sistema PLC

Per i sistemi di elaborazione è importante stabilire un modello valido che sia coerente e allo stesso tempo adattabile alle modifiche, in modo da assicurare una soluzione efficiente e affidabile per l'analisi di sistemi complessi.
In questo caso, il modello oggetto di studio è un sistema PLC nelle sue componenti hardware e software, valide per tutti i sistemi di questo tipo.
Nel modello proposto di seguito, l'architettura e le funzionalità dei PLC vengono modellate come componenti separate tramite automi a stati finiti con l'aggiunta di variabili e porte.
Questa tecnica di modellazione è detta BIP (Behaviour, Interaction, Priority) e permette di rappresentare il funzionamento delle singole componenti, oltre che l'interazione e la sincronizzazione tra le parti.
Nei modelli BIP, le variabili sono utilizzate per memorizzare dati locali, mentre le porte rappresentano le istruzioni (coincidono con le etichette delle transizioni) che vengono utilizzate per l'interazione tra le componenti. Ad ogni stato corrisponde un'attesa del sistema di un evento che ne causi la transizione ad un altro stato.
Prima di modellare le componenti, si fornisce il modello d'insieme dell'architettura del PLC. Tale modello si divide in tre parti: hardware, software e ambiente circostante, ovvero segnali esterni e dispositivi di ingresso e uscita a cui il PLC è collegato. In figura 4.6 è visibile la separazione tra i livelli.

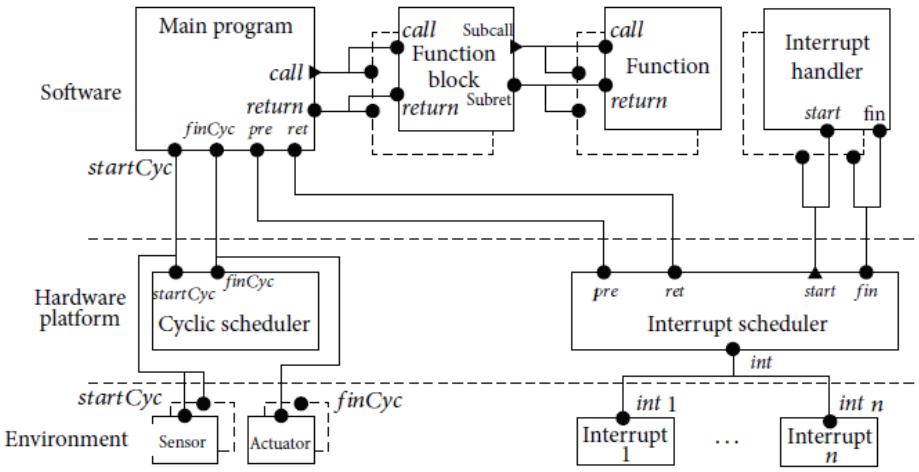

Figura 4.6- Architettura di un sistema PLC, [5]

Al primo livello è descritta la parte software, nelle componenti main, blocchi funzionali (FB), funzioni (FC) e interrupt handler.

Il programma main può richiamare FC o FB tramite la porta *call*, dalla quale viene inviato un segnale di broadcast che compara il nome del blocco da eseguire con i nomi dei blocchi in memoria, eseguendo il blocco corretto. A loro volta i blocchi funzionali possono richiamare funzioni (*Subcall)*.

In maniera inversa, al termine dell'esecuzione del proprio codice, i blocchi restituiscono il risultato delle proprie operazioni tramite la porta *return.*

Al secondo livello viene descritta la piattaforma hardware che si occupa dell'esecuzione ciclica e della gestione degli interrupt. Le componenti in questione sono lo scheduler ciclico e lo scheduler interrupt.

Infine, a livello ambiente vengono descritti i dispositivi controllati. Ad esempio, il PLC legge l'informazione proveniente dai sensori ad ogni esecuzione del ciclo CPU tramite la porta *startCyc.* Al termine dell'esecuzione del programma, vengono comandati gli attuatori collegati sulle uscite tramite la porta *finCyc.*

I segnali di interrupt godono invece di una gestione a parte tramite apposite componenti.

4.4.1 Ciclo CPU

È stato illustrato come il ciclo di esecuzione CPU di un PLC consista nella ripetizione delle fasi di lettura dei segnali, esecuzione delle istruzioni e aggiornamento delle uscite.

In figura 4.7 è modellato tale funzionamento tramite la formalizzazione dello scheduler ciclico, responsabile di questa procedura.

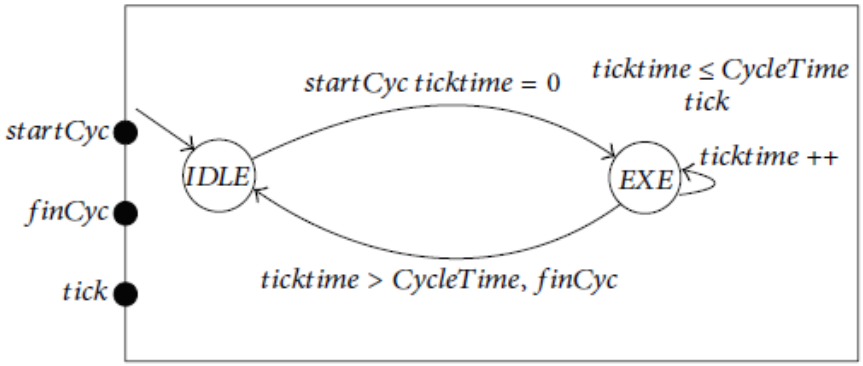

Figura 4.7 - Modello dello scheduler ciclico, [5]

Inizialmente ci si trova nello stato di riposo IDLE. All'avvio del programma main (*startCyc*), si passa allo stato di esecuzione EXE, in cui si rimane ad ogni segnale di clock (*tick*), fintanto che vengono eseguite le istruzioni del codice utente.

Terminata l'esecuzione dopo un tempo *CycleTime*, si ritorna allo stato IDLE tramite la sincronizzazione sulla porta *finCyc.* L'esecuzione continua ciclicamente.
Si noti che lo stato IDLE accorpa la prima e l'ultima fase del ciclo ed è responsabile della sincronizzazione con l'ambiente, ovvero con i dispositivi di ingresso e uscita.

4.4.2 Blocco timer

Con la stessa tecnica utilizzata nella descrizione del ciclo CPU, si possono modellare le componenti hardware e software, ma anche funzionalità specifiche del programma.
Prendiamo ad esempio un blocco timer di tipo TON, con funzione ingresso-uscita descritta in figura 4.8:

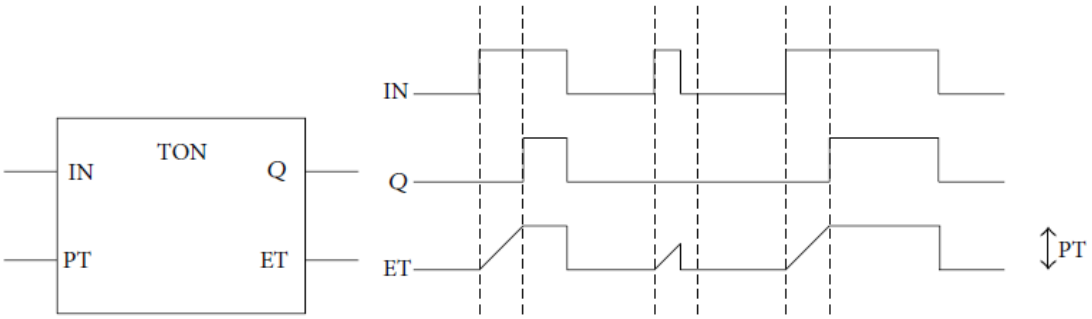

Figura 4.8 - Timer TON, [5]

Il modello è quindi descritto in figura 4.9 in tre stati: IDLE (riposo), BUSY (timer avviato) e TIMEOUT (temporizzazione terminata).
L'ingresso di abilitazione (IN) è modellato tramite le porte *set* e *reset,* mentre l'evento *readQ* fornisce l'informazione sullo stato dell'uscita.

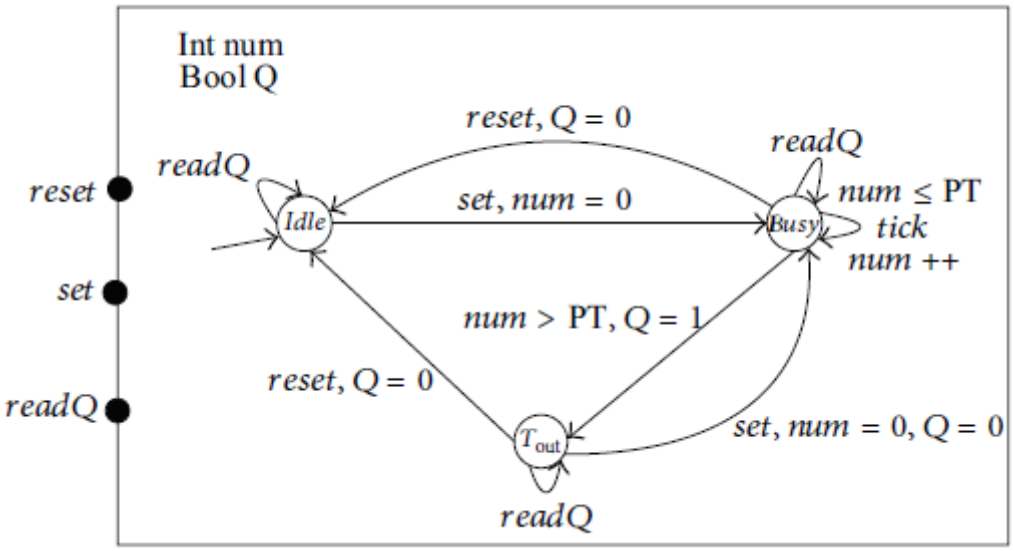

Figura 4.9 - Modello di un timer TON, [5]

Dallo stato iniziale, il segnale *set* attiva la transizione verso lo stato BUSY, resettando la variabile *num.* Quest'ultima immagazzina l'informazione sul tempo trascorso dall'attivazione del timer e viene incrementata ad ogni segnale *tick*.

Al raggiungimento del valore preset (PT) si passa allo stato TIMEOUT e viene settata l'uscita (Q).

Terminata la modellazione di blocchi hardware e software e delle chiamate delle varie funzioni, si uniscono le componenti andando a formare l'architettura vista nella sua globalità in figura 4.6.

La modellazione può essere arricchita formalizzando le istruzioni eseguibili attraverso tecniche quali modelli di Kripke [5].

Nonostante non siano state impiegate nello sviluppo di questa tesi, le tecniche di modellazione per sistemi di controllo risultano essere una soluzione efficace per garantire il corretto funzionamento e la sicurezza di sistemi anche complessi.

Software in grado di modellare e controllare questo tipo di sistemi possono essere applicati per la verifica della corretta progettazione dell'hardware, del software e dei protocolli di comunicazione.

Capitolo 5
Software di sviluppo

Messersì Supervisor consiste nello sviluppo di due applicazioni, desktop e mobile, le cui maggiori differenze derivano dalle piattaforme di sviluppo. Vengono quindi illustrati ambienti, piattaforme e software impiegati, sia per le componenti comuni che per quelle caratterizzanti.

5.1 Visual Studio

Microsoft Visual Studio è l'ambiente di sviluppo utilizzato per la scrittura del codice. Fornisce un IDE che ne facilita non solo la scrittura, ma anche il debug e il rilascio al termine dello sviluppo.

Figura 5.1 - Microsoft Visual Studio

Sono supportate diverse tipologie di applicativi per varie piattaforme. Dalla schermata iniziale basta selezionare "Nuovo progetto" alle voci "App WPF" (figura 5.2) o "Xamarin.Forms".

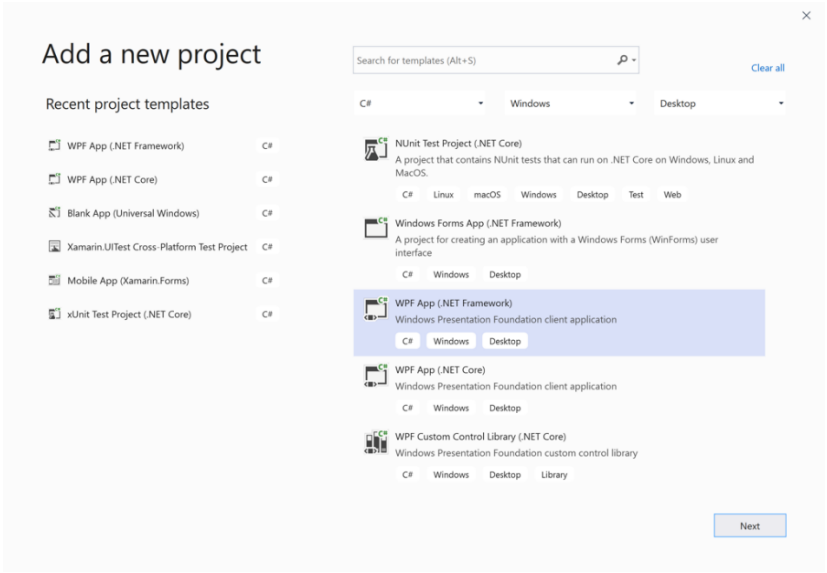

Figura 5.2 - Nuovo progetto per app WPF

5.2 Microsoft .NET

Figura 5.33 - .Net Framework

Con .NET Framework si intende la piattaforma di sviluppo software sviluppata dall'azienda statunitense Microsoft per la progettazione di vari tipi di applicativi. La sua peculiarità è l'essere indipendente dalla versione operativa di Windows su cui opera.

Le applicazioni che rientrano in questa piattaforma godono di svariate funzionalità espressamente progettate per integrarsi in Internet e garantire il massimo grado di sicurezza e integrità dei dati.

Il framework .NET riunisce un gran numero di soluzioni per la gestione della memoria e vari servizi di sistema, oltre a un'ampia libreria di classi che consente ai programmatori di sfruttare varie funzionalità per lo sviluppo delle principali tipologie di applicazioni.

Sono supportati diversi linguaggi (C#, Visual Basic, F#), editor e librerie per creare Web App, applicazioni mobile, desktop e giochi.

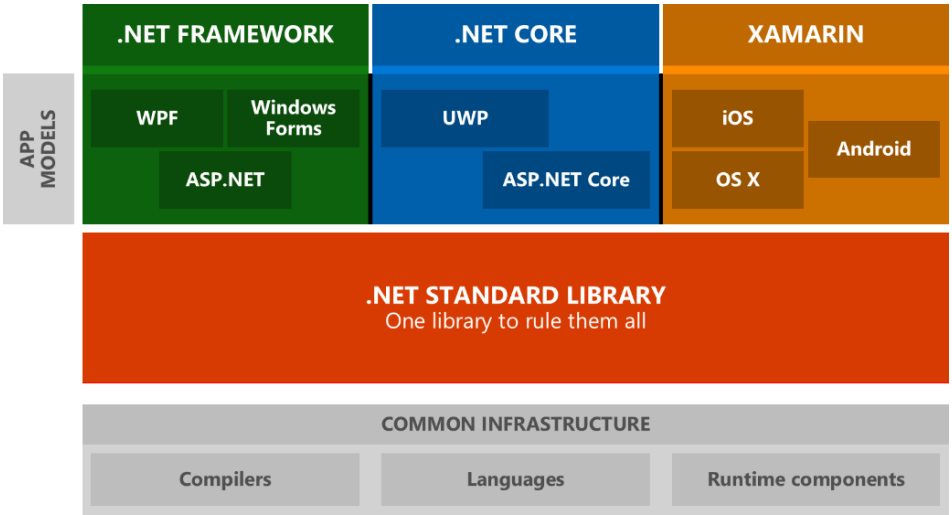

Figura 5.44 - Componenti del framework .NET

Il codice viene eseguito grazie al *Common Language Runtime*, l'environment per la macchina virtuale e le librerie standard della piattaforma .NET, che fornisce anche le funzioni di compilatore e debugger.

Inoltre, Microsoft e sviluppatori di terze parti collaborano nell'ambiente .NET rilasciando pacchetti, detti NuGet, specifici per .NET. Un pacchetto NuGet può essere scaricato e inserito

nella soluzione per implementare particolari funzionalità. Attualmente sono disponibili più di 90 mila pacchetti[12].

Per lo sviluppo dell'applicazione desktop si è utilizzata la versione .NET 4.6.1. L'app Android è stata invece sviluppata sulla piattaforma Xamarin ed è quindi basata sul progetto Mono, parte di .NET Standard 2.0, nato come implementazione del framework .NET per ambienti non Windows.

5.2.1 Linguaggi di programmazione

Sono due i linguaggi impiegati nello sviluppo: il linguaggio C-Sharp (C#) e il linguaggio XAML. Questo perché entrambi le versioni hanno una struttura di tipo GUI (Graphical User Interface), in cui le applicazioni combinano elementi grafici, detti *markup*, (etichette, textbox, pulsanti, ecc.) con una parte di codice, in questo caso scritta in C#, detta *Code-Behind*, che appunto si pone "dietro" la parte grafica con cui l'utente interagisce.

Il C-Sharp o C# è un linguaggio di programmazione ad alto livello orientato agli oggetti, creato e sviluppato insieme alla piattaforma .NET all'inizio degli anni 2000 come risposta di Microsoft alla diffusione di Java. C# ereditò molte delle caratteristiche di linguaggi già esistenti quali C++, Visual Basic e Java stesso.

Essendo un linguaggio orientato agli oggetti, in C# si definiscono classi e oggetti in grado di interagire gli uni con gli altri attraverso lo scambio di messaggi.

XAML (EXtensible Application Markup Language) è invece un linguaggio basato su XML e progettato per la creazione di elementi dell'interfaccia grafica e l'inizializzazione di oggetti della vista. Definire gli oggetti in XAML, consente di ottenere un codice molto più pulito, evitando di sovraccaricare il codice C# associato all'interfaccia.

```xaml
<Label Text="Hello from XAML!"
       IsVisible="True"
       Opacity="0.75"
       HorizontalTextAlignment="Center"
       VerticalOptions="CenterAndExpand"
       TextColor="Blue"
       BackgroundColor="#FF8080"
       FontSize="Large"
       FontAttributes="Bold,Italic" />
```

```csharp
new Label
{
    Text = "Hello from Code!",
    IsVisible = true,
    Opacity = 0.75,
    HorizontalTextAlignment = TextAlignment.Center,
    VerticalOptions = LayoutOptions.CenterAndExpand,
    TextColor = Color.Blue,
    BackgroundColor = Color.FromRgb(255, 128, 128),
    FontSize = Device.GetNamedSize(NamedSize.Large, typeof(Label)),
    FontAttributes = FontAttributes.Bold | FontAttributes.Italic
};
```

Figura 5.5 - Istanza di un controllo Label in XAML e in code-behind C#

Le interfaccio scritte nel file XAML vengono analizzate durante compilazione ed esecuzione, in modo da individuare e definire i collegamenti tra questi oggetti e le istanze nel codice C#.

[12] https://dotnet.microsoft.com/learn/

Per la progettazione di ogni pagina o finestra sono quindi necessari un documento XAML e il suo corrispondente file in C#, poiché il markup scritto in XAML necessita del supporto del code-behind per la gestione degli eventi e l'interazione con i controlli.

5.3 Applicazioni WPF

La prima applicazione sviluppata, quella in versione desktop, si basa su WPF (Windows Presentation Foundation), il più recente approccio Microsoft per lo sviluppo delle interfacce utente all'interno del framework .NET.

L'unico requisito di funzionamento di un'applicazione WPF è l'installazione del framework .NET. Tuttavia, poiché Microsoft include .NET in ogni versione dei sistemi operativi successi a Windows Vista, la maggior parte degli utenti è in grado di eseguire questo tipo di applicazione.

In WPF è offerta la base operativa per l'esecuzione delle applicazioni Windows con una struttura che riesce a separare la programmazione della parte grafica dal codice logico.

Allo stesso tempo, le due componenti interagiscono tra loro con il meccanismo del Data-Binding: grazie ad una apposita classe si possono legare dinamicamente proprietà di un di un controllo (target) a quelle di un oggetto (source).

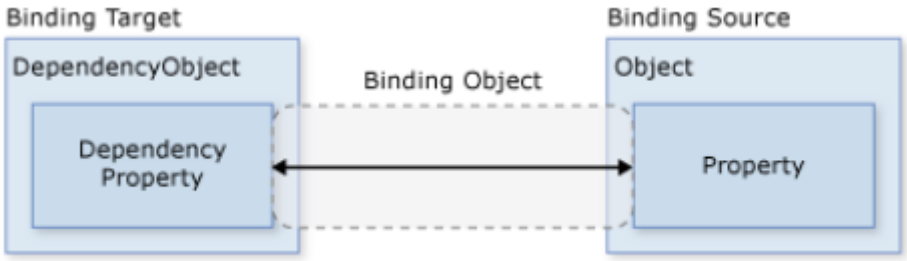

Figura 5.6 - Schema del Data-Binding, [7]

Le classi sono quindi definite parziali, poiché le due parti vengono combinate in fase di Runtime tramite il metodo InitializeComponent().

Ad esempio, per istanziare un oggetto di tipo Window, WPF mette a disposizione un file .xaml e un file .cs, inizializzando una finestra predefinita simile a quella in figura 5.7.

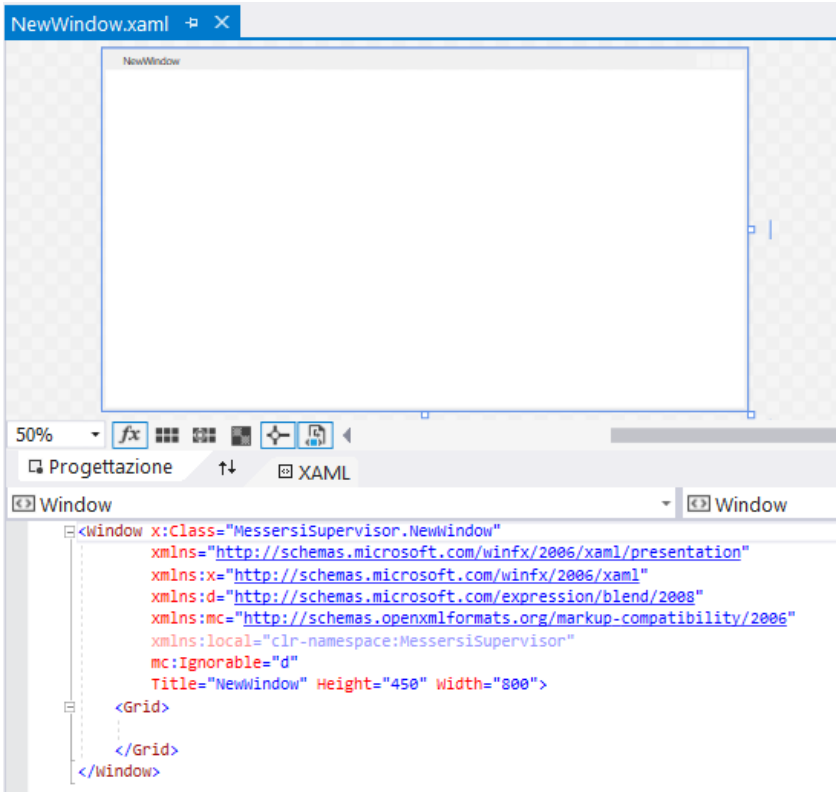

Figure 5.75 - Codice XAML di una nuova finestra per app WPF

Nel file XAML, l'attributo x:class informa la parte grafica sul code-behind utilizzare. Per quanto riguarda l'esempio sopra citato, in figura 5.8 è riportata la corrispondente classe descritta nel file NewWindow.xaml.cs.

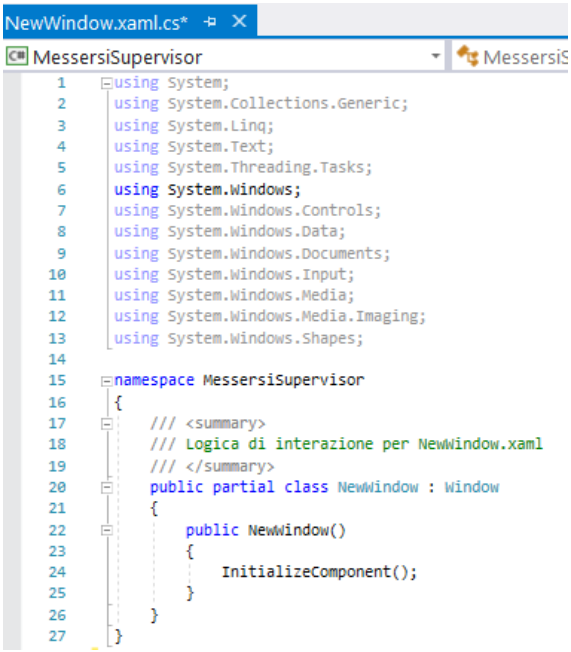

Figura 5.8- Code-behind per una nuova finestra WPF

Alla creazione del progetto vengono inoltre generati un file App.xaml e App.xaml.cs, che comporranno l'oggetto di tipo Application, punto di inizio dichiarativo dell'applicazione.

Nel file App.xaml sono definite le risorse le globali, le eccezioni non gestite e soprattuto la proprietà StartUpUri, indicante la finestra o pagina con cui si avvierà l'applicazione. Nell'esempio di figura 5.9 l'esecuzione si avvierà dal file MainWindow.xaml.

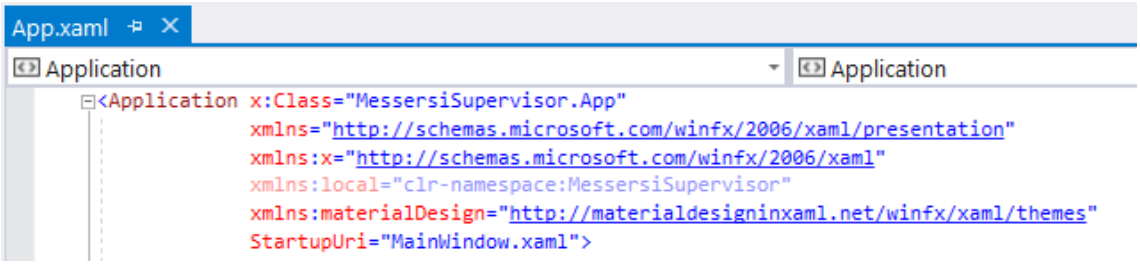

Figura 5.9 - File App.xaml di MessersiSupervisor

5.4 Xamarin

L'applicazione mobile è stata realizzata sempre in Visual Studio, ma sulla piattaforma Xamarin, software di progettazione di applicazioni native e multipiattaforma del framework .NET.
Sono due gli approcci disponibili: utilizzare Xamarin.Forms per progetti cross-platform o implementare le SDK native (*Software Development Kit*, pacchetti di sviluppo applicazioni) utilizzando Xamarin.Android o Xamarin.iOS. Si è optato per la prima soluzione per avere una portabilità più ampia del codice.

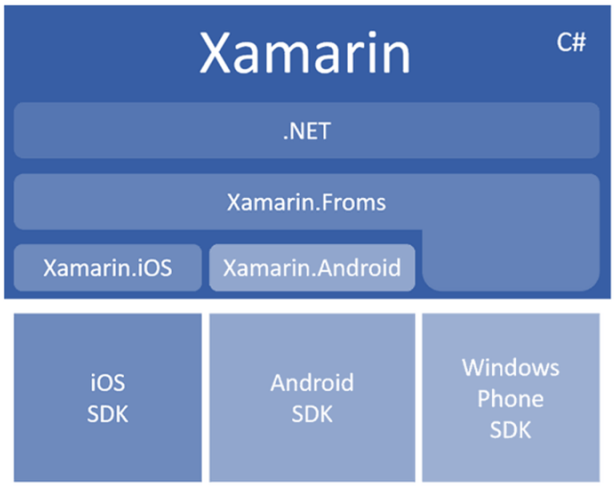

Figura 5.10 – Componenti di Xamarin e approccio alle UI

Il suo grande vantaggio è la possibilità di descrivere un'unica interfaccia grafica (UI) e un unico codice in C# valido per diverse piattaforme (iOS, Android, Windows Phone, ecc.), ma con la possibilità di mantenere per ogni piattaforma elementi, funzionalità e codice specifici.

Per la costruzione dell'interfaccia grafica, Xamarin.Forms fornisce elementi come pagine, layout e controlli, descritti con classi parziali come in WPF, tramite XAML e codice C#.
Creando una nuova soluzione Xamarin.Forms viene generato un progetto portable per la logica condivisa e la definizione della UI, oltre a vari progetti contenente le versioni native di ogni piattaforma.
All'interno del progetto portable è presente la coppia di file App.xaml e App.xaml.cs che crea l'oggetto di tipo Application da cui partirà l'esecuzione. La sintassi e il funzionamento di questi file sono pressoché gli stessi descritti per le applicazioni WPF, con differenze non significative.

5.5 Libreria S7.Net

Il driver per le istruzioni per lo scambio dati con i PLC è la libreria open source S7.Net, la quale mette a disposizione le istruzioni C# necessarie all'interfacciamento a PLC dotati di connessione Profinet.
S7.Net è compatibile con il framework .NET e con i PLC Siemens della serie S7, ovvero i dispositivi S7-200, Logo 0BA8, S7-300, S7-400, S7-1200, S7-1500.
La connessione avviene istanziando oggetti della classe Plc, nella quale sono definiti attributi e metodi per la comunicazione.
Il costruttore della classe Plc richiede parametri quali il modello CPU, l'indirizzo IP e i numeri di rack e slot. A questo punto è possibile aprire la connessione tramite il metodo Open(). È possibile mantenere connessioni con più di un PLC contemporaneamente e da più applicativi allo stesso PLC.

```
public partial class NewWindow : Window
{
    Plc myplc = new Plc(CpuType.S71200, "192.168.25.150", 0, 0);

    public NewWindow()
    {
        InitializeComponent();

        // Apertura connessione con il PLC
        myplc.Open();
    }
}
```

Figura 5.11 - Istanza dell'oggetto Plc e connessione

Prima di eseguire ogni istruzione è buona regola controllare che la connessione sia attiva tramite la proprietà IsConnected. Questo per evitare blocchi del programma nel caso la connessione Ethernet si interrompa.

Sono disponibili vari metodi per la lettura e la scrittura di variabili del PLC: si può accedere ai valori di ingressi, uscite, DB dati, timer e contatori. Inoltre, è possibile leggere e scrivere singoli bit, bytes o strutture su richiesta dell'utente.

```
if (myplc.IsConnected)
{
    // Lettura del bit DBX0.0 della DB1
    bool valore_lettura = (bool)myplc.Read("DB1.DBX0.0");

    // Settaggio a true dello stesso bit
    myplc.Write("DB1.DBX0.0", true);
}
```

Figura 5.12 - Esecuzione di istruzioni Read e Write

Infine, per i dispositivi S7-1200 e S7-1500 è necessario abilitare l'accesso da remoto nel PLC. Nelle opzioni di protezione della CPU va consentito l'accesso completo a partner remoti (figura 5.13) e va disattivato l'accesso ottimizzato agli attributi delle DB con cui si vuole interagire.

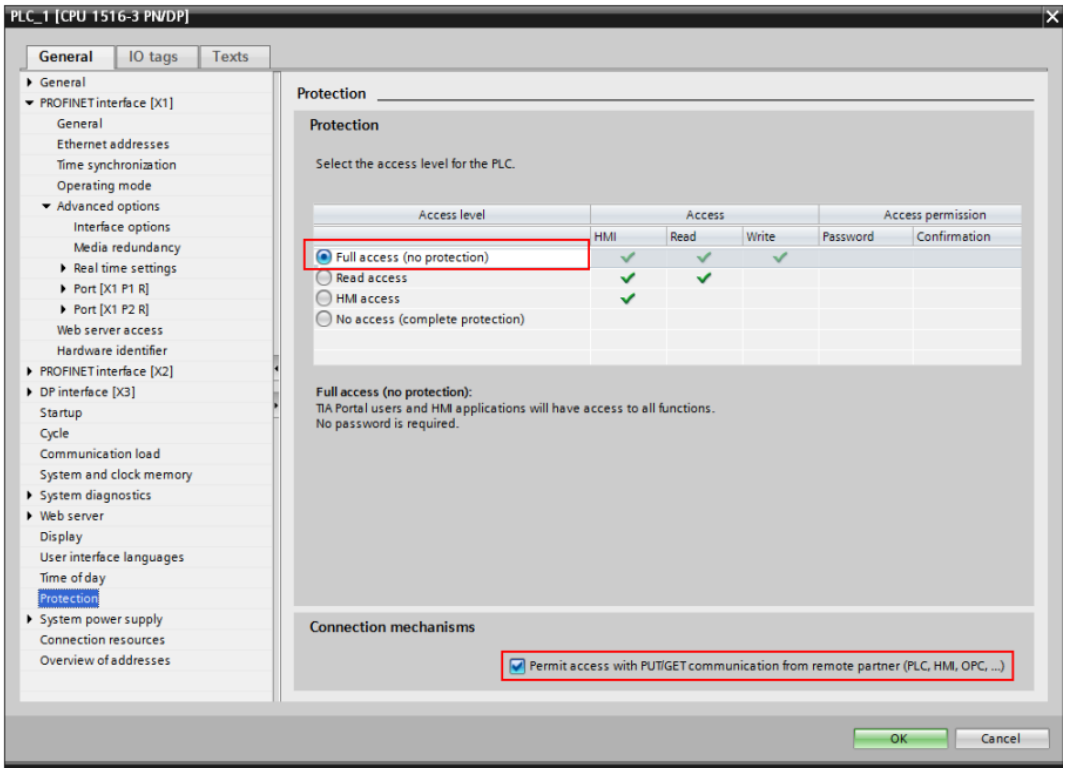

Figura 5.13 - Configurazione della CPU del PLC per l'accesso remoto, S7.Net documentation

5.6 Database e SQL

Il funzionamento dell'app Messersì Supervisor è strettamente legato all'accesso a informazioni contenute in un database di tipo relazionale. Esso è posto sul server aziendale in modo da essere accessibile solo all'interno della rete o da remoto previa verifica delle credenziali.

I database costituiscono archivi per la memorizzazione e la gestione dati tramite i quali si possono ricercare, aggiungere, modificare o cancellare informazioni.
In questo modo l'applicativo non deve occuparsi dell'archiviazione delle informazioni, ma solo dell'elaborazione. Inoltre, viene consentito l'utilizzo del supervisore a più utenti anche simultaneamente.
Nei database relazionali, i dati sono organizzati in tabelle i cui campi (le colonne) possono ospitare vari tipi di dato (stringhe, interi, date, immagini, ecc.). Tra i campi è necessaria una chiave primaria, che identifica univocamente ogni riga della tabella.
È poi possibile realizzare relazioni tra due o più campi di diverse tabelle, ovvero definire relazioni che legano tra loro righe appartenenti a tali campi. In questo caso si parla di chiavi esterne.

5.6.1 Database MessersiApp

Per realizzare il database sono stati utilizzati gli strumenti offerti dalla piattaforma XAMPP, software che mette a disposizione servizi come il web server Apache e il gestore di database MySQL. Grazie a queste due componenti si ottiene un server locale che funge da host per il database. Quest'ultimo è scrivibile dall'applicazione web phpMyAdmin, anch'essa parte dei servizi di XAMPP.
In questo modo è stato definito il database MessersiApp, al cui interno state definite le seguenti tabelle e le relazioni tra esse.

- Utenti: necessaria per l'identificazione del personale autorizzato ad accedere all'applicazione. Per ogni username viene anche indicato un indirizzo di posta elettronica dal quale viene inviata l'e-mail di notifica in caso di segnalazione allarmi.
- Modelli_PLC: contiene modelli e brand (attualmente solo Siemens) dei PLC ai quali è possibili connettersi.
- Macchine: archivia i dati delle macchine alle quali è possibile connettersi.
- ConvModel: assegna un particolare gruppo ad ogni modello macchina. Sulla base di tale gruppo è gestita poi la lettura degli allarmi. Infatti, gli indirizzi delle DB di allarme variano a seconda della tipologia di macchina (reggiatrici orizzontali, verticali, incappuciatrici, ecc.).
- Allarmi: fornisce all'applicazione gli indirizzi e i testi degli allarmi di ogni gruppo macchina.

In figura 5.14 è illustrata la struttura del database in maniera grafica. È quindi possibile notare le relazioni tra le tabelle e le chiavi esterne.

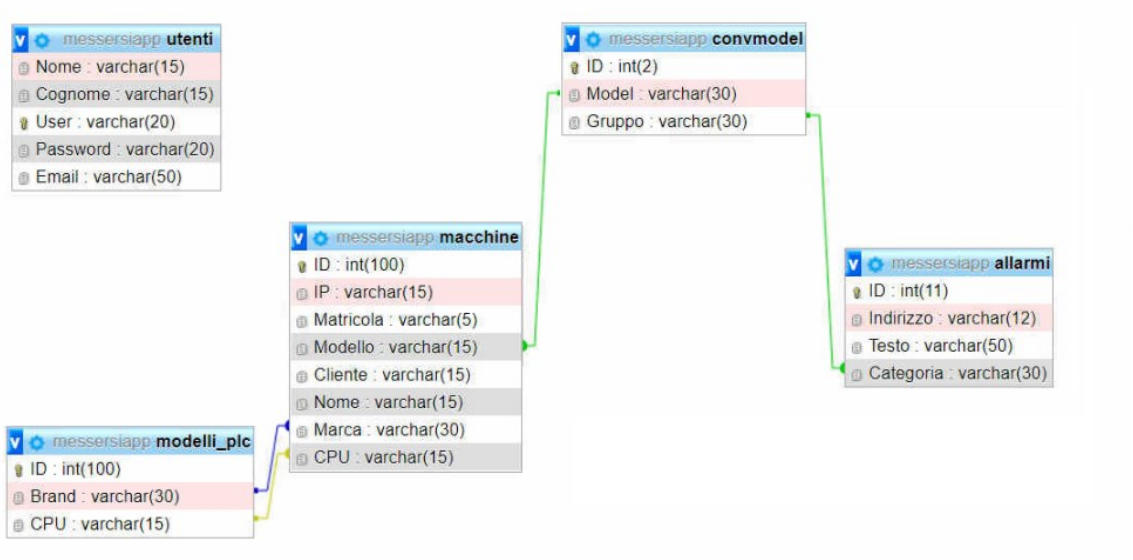

Figura 5.14 – Struttura del database MessersiApp

5.6.2 Libreria MySQL

Le operazioni eseguibili su un database vengono chiamate *query* e possono essere di due tipi: interrogative, quali ricerca e selezione, o di manipolazione, quali inserimento, modifica e cancellazione di elementi.

Queste istruzioni vengono eseguite nel linguaggio SQL (Structured Query Language), sviluppato appositamente per definirne la sintassi.

Nelle applicazioni del framework .NET è possibile connettersi a database ed eseguire query in SQL implementando le istruzioni della libreria MySql.

Va innanzitutto definita una stringa per la connessione, contenente l'indirizzo del server ospitante, il nome del database e le credenziali di accesso al servizio phpMyAdmin.

```
// Variabili per connessione database SQL
static string server = "192.168.0.19";//IP macchina virtuale
static string database = "messersiapp";
static string uid = "messersi";
static string password = "*****";
static string connectionString = "SERVER=" + server + ";" + "DATABASE=" + database + ";"
                                 + "UID=" + uid + ";" + "PASSWORD=" + password + ";";
```

Figura 5.15 - Variabili per la connessione al database

Vanno poi istanziati un oggetto di tipo MySqlConnection per aprire la connessione e un oggetto MySqlCommand per l'esecuzione della query. In figura 5.16 è illustrato il codice per leggere e mostrare su una data grid le righe della tabella macchine.

```
try
{
    connection = new MySqlConnection(connectionString);
    MySqlCommand cmd = new MySqlCommand("SELECT * FROM macchine", connection);
    connection.Open();
    DataTable dt = new DataTable();
    dt.Load(cmd.ExecuteReader());
    dtGridMacchine.DataContext = dt;
}
catch (MySqlException exc) { MessageBox.Show(exc.Message); }
finally
{
    connection.Close();
}
```

Figura 5.16 - Codice per la query di lettura della tabella macchine

Il blocco try-catch è sempre necessario per evitare che il verificarsi di eccezioni durante la connessione al database blocchino l'esecuzione dell'applicazione.

Per ottenere un risultato efficiente è stato quindi necessario integrare tutti questi ambienti e piattaforme, mantenendo una programmazione snella e funzionale.
Terminata la descrizione delle tecnologie impiegate, nel prossimo capitolo verrà illustrato e commentato il risultato del codice prodotto.

Capitolo 6
Funzionamento di Messersì Supervisor

L'architettura dell'applicazione sviluppata, le scelte progettuali adottate e il funzionamento dell'applicazione sono illustrati in questo capitolo dedicato al risultato ottenuto.

6.1 Architettura del programma

Lo sviluppo di applicazioni WPF e Xamarin.Forms è basato su pattern View-Model, ovvero su un'architettura in cui i template XAML vengono messi in relazione con gli oggetti del modello e le loro proprietà e metodi.

Basandosi su questo approccio, sono state quindi sviluppate finestre e pagine che includano i controlli tramite il quale l'utente esegue istruzioni che vengono elaborate dal programma.

Le versioni desktop e mobile implementano quasi totalmente le stesse funzionalità, riassunte nel diagramma dei casi d'uso:

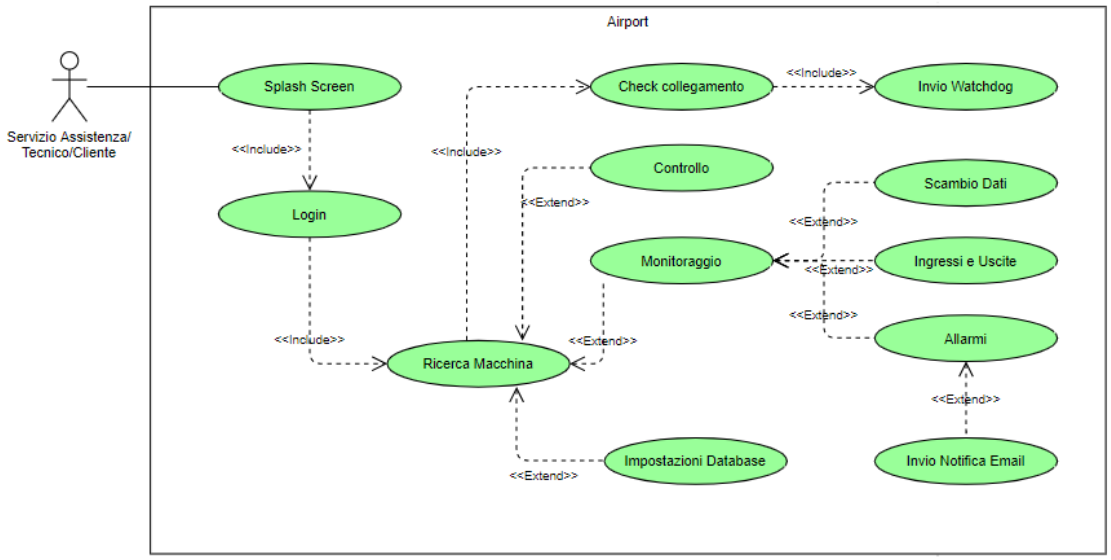

Figura 6.1 - Diagramma dei casi d'uso

Per ogni pagina viene istanziato un oggetto della corrispondente classe di tipo parziale, poiché definisce il markup nel file .xaml e la logica di funzionamento nel code-behind.

Nel *class diagram* di figura 6.2 sono raccolte le classi utilizzate nel codice per la versione WPF, rimaste comunque sostanzialmente invariate per la versione mobile.

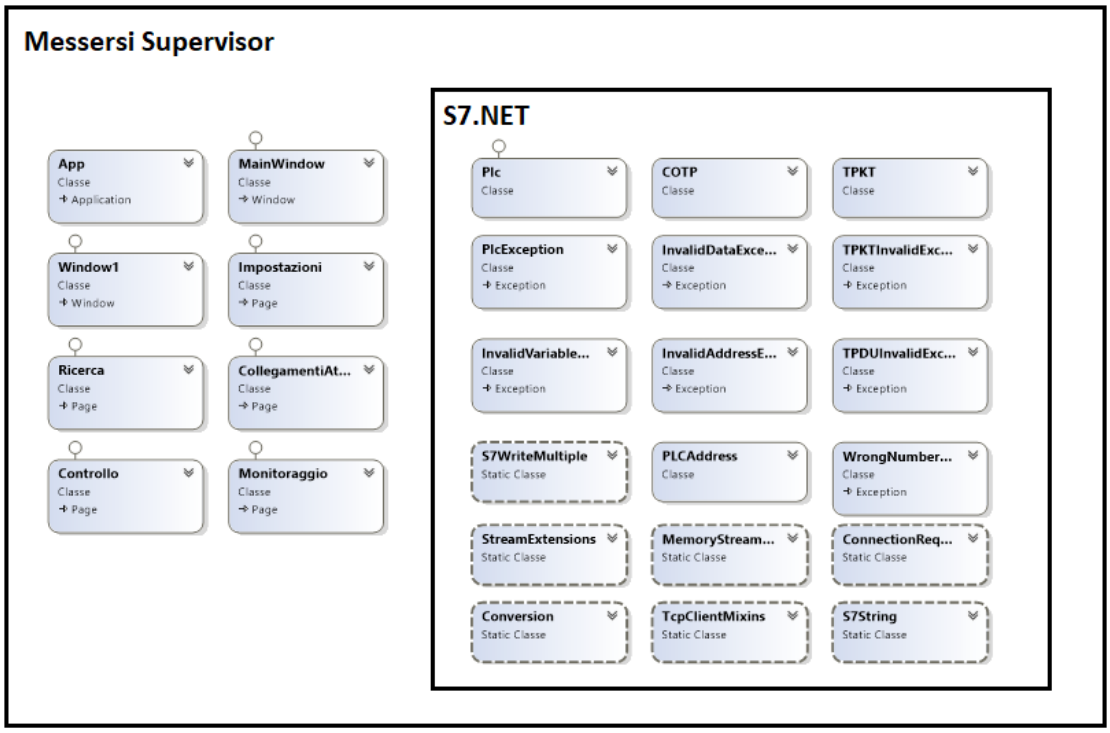

Figura 6.2 - Class diagram

Di seguito saranno descritte le funzionalità principali di tali classi e le loro implementazioni.

6.2 Login

La MainWindow è la classe di avvio inizializzata nel file app.xaml, nella quale l'utente si trova a dover immettere le proprie credenziali nella pagina di login per poter entrare nell'applicazione.

Figura 6.3 - Classe MainWindow

Si tratta di una classe ereditata da Window, in cui vengono semplicemente definiti i parametri per la connessione al database (identici per le pagine seguenti) e i metodi per la verifica delle credenziali.

Figura 6.4 - Pagina di login

Nella versione mobile, durante l'apertura dell'applicazione viene mostrata una pagina di caricamento, in gergo splash screen.

Figura 6.5 - Splash screen

Se l'utente viene autorizzato ad accedere, vengono passati alla classe Impostazioni i dati dell'utente che ha eseguito il login, in modo da predisporne una eventuale modifica delle credenziali.

Successivamente, viene chiusa la MainWindow e si istanzia una nuova finestra Window1 il cui contenuto è navigabile tramite i pulsanti posti nell'apposito menù laterale. Analogamente, l'app Android utilizza una pagina di tipo Master-Detail per la navigazione.

Figura 6.6 - Classe Window1

Le classi descritte in seguito saranno oggetti di tipo Page, visualizzate come contenuto della finestra Window1.

Figura 6.7 – Menù di navigazione

6.3 Ricerca macchina

Dopo il login si viene indirizzati alla pagina di ricerca, dalla quale si può selezionare una macchina registrata nel database per avviare un tentativo di connessione. Si possono modificare i parametri di rack e slot del PLC.

Tutte le pagine implementano una lista contenente i PLC connessi e un oggetto Plc corrispondente al PLC da ricercare, controllare o monitorare, in modo che eventuali nuove connessioni o disconnessioni vengano sempre comunicate.

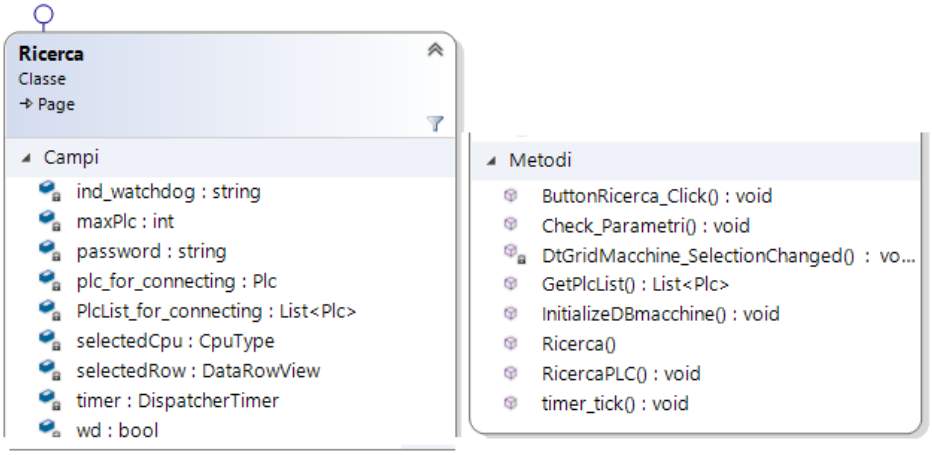

Figura 6.8 - Classe Ricerca

- ind_watchdog: indirizzo per la scrittura del watchdog nella DB di invio dati. A tale scopo viene utilizzato un timer e il bit ausiliare wd.
- maxPlc: numero massimo di PLC connessi simultaneamente.
- selectedCpu: modello di CPU selezionato per la connessione, ricavato dalla riga selezionata (selectedRow) nel data grid che mostra la tabella macchine del database.
- plc_for_connecting: istanza del PLC di cui si effettuerà la connessione.
- PlcList_for_connecting: lista dei PLC attualmente connessi.
- ButtonRicerca_Click(): metodo generato alla pressione del pulsante di ricerca, verifica i parametri immessi per il PLC e, se i valori sono ammissibili, ne avvia la connessione tramite il metodo RicercaPLC().
- InitializeDBMacchine(): legge i valori presenti nella tabella macchine del database e li mostra nell'apposita data grid.
- DtGridMacchine_SelectionChanged(): alla modifica della selezione del data grid macchine vengono passati i parametri necessari al costruttore dell'oggetto Plc.
- CheckParametri(): impedisce valori non ammessi di rack slot in base alla CPU della macchina selezionata.
- GetPlcList(): metodo tramite il quale le altre pagine accedono alla lista dei PLC connessi.

Figura 6.9 - Pagina di ricerca

Se si ha almeno un PLC attualmente connesso, il bordo delle pagine viene evidenziato in verde.

6.4 Connessioni attive

Qui vengono mostrate le informazioni riguardanti la macchina connessa e selezionata. La spunta verde mostra che il collegamento è attivo, mentre l'icona del PLC lampeggiante indica la ricezione del segnale di watchdog.

La corrispondente classe è rinominata CollegamentiAttivi e implementa le funzioni che permettono la visualizzazione delle informazioni sulla macchina connessa e l'eventuale disconnessione.

Figura 6.10 - Classe CollegamentiAttivi

- plc_to_check: istanza del PLC di cui si effettua la lettura delle informazioni.
- selectedPlc: utilizza i metodi GetSelectedPlc() e SetSelectedPlc() viene mantenuta la selezione del PLC corrente per le altre pagine.
- timer: utilizzato per eseguire ogni mezzo secondo la lettura del watchdog, visibile dal lampeggio dell'icona del PLC ottenuto con il metodo ToggleImage().
- ButtonDisconnectClick(): esegue la disconnessione dal PLC selezionato.
- checkConnection(): aggiorna la visualizzazione dei dati riguardanti la macchina selezionata.

Figura 6.11 - Pagina delle connessioni attive

6.5 Controllo

Dalla pagina di controllo possono essere comandati i valori nella DB di scambio dati o qualunque altra memoria immettendone l'indirizzo e specificando il valore di comando.

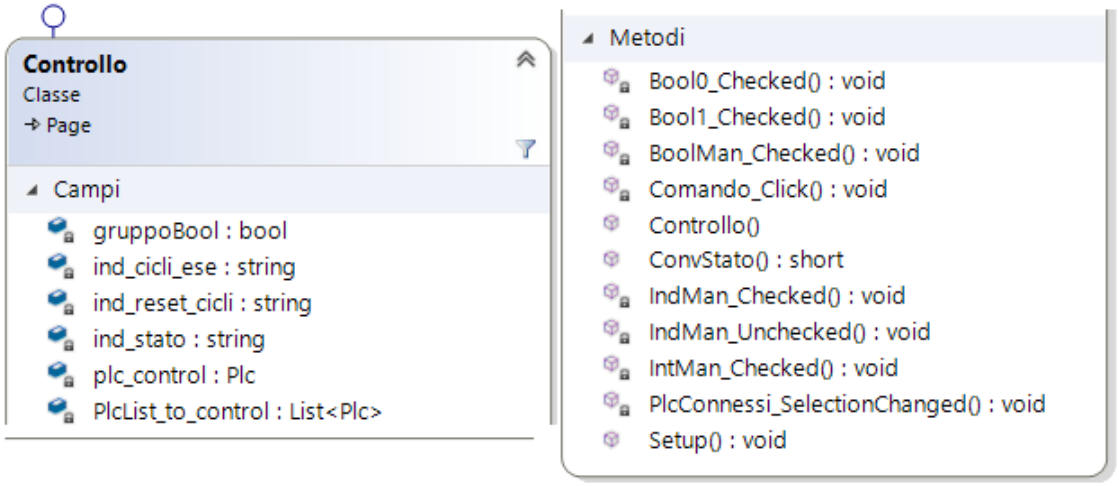

Figura 6.12 - Classe Controllo

- plc_to_control: istanza del PLC da controllare.
- gruppoBool: contiene il valore false o true selezionato per il controllo di variabili bool
- Setup(): aggiorna le opzioni abilitate in base al tipo di dato selezionato. Comandando una variabile bool si utilizzano le opzioni 0/1, con variabile intere si abilita l'apposito textbox, mentre per l'impostazione dello stato sono presenti le opzioni specifiche. Quando selezionata, ciascuna opzione genera il corrispondente evento checked().
- ConvStato(): converte la modalità di funzionamento macchina selezionata con il corrispondente valore intero da inviare nella corrispondente DB di scambio dati.

Per ragioni di sicurezza è stata inibita la possibilità di comandare direttamente le uscite del PLC.

Figura 6.13 - Pagina di controllo

6.6 Monitoraggio

Analogamente al controllo, nella pagina di monitoraggio vengono mostrati i valori letti nella DB di scambio dati o, in alternativa, si può eseguire la lettura degli allarmi o degli ingressi e uscite. Nella classe Monitoraggio vengono definite le variabili ausiliarie per le letture, eseguite e visualizzate tramite gli specifici metodi.

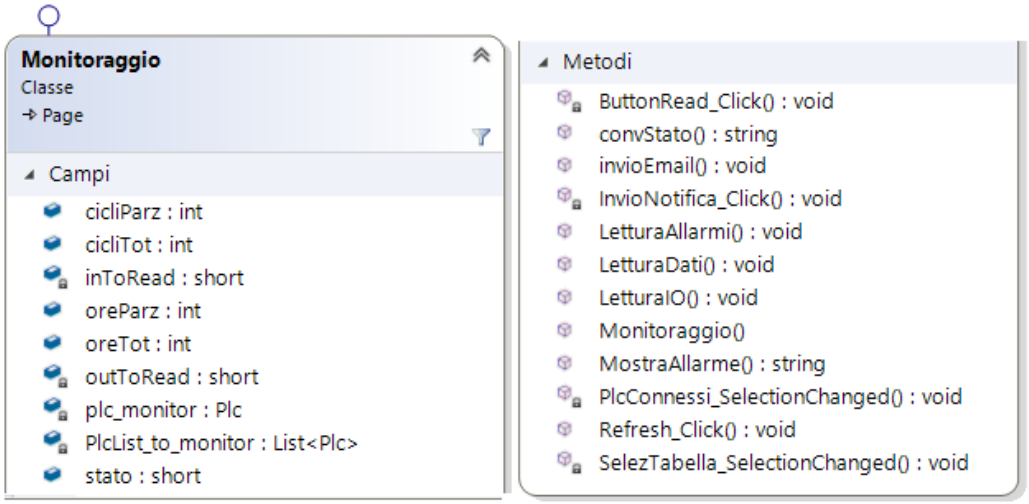

Figura 6.14 - Classe Monitoraggio

- plc_monitor: istanza del PLC del quale visualizzare i dati.
- inToRead e outToRead: indica il numero di ingressi e uscite di cui effettuare la lettura a partire dagli indirizzi I0.0 e Q0.0.
- LetturaDati(): esegue la lettura dei dati inviati dal PLC sulla DB101, ovvero delle informazioni riguardanti cicli eseguiti, ore di lavoro e stato della macchina.
- LetturaIO(): effettua la scansione di ingressi e uscite del PLC selezionato.
- ButtonRead_Click(): permette la lettura di un qualsiasi indirizzo di memoria inserito nel textbox in basso a sinistra nella pagina.
- Refresh_Click(): aggiorna la lettura del plc_monitor per la tabella selezionata.

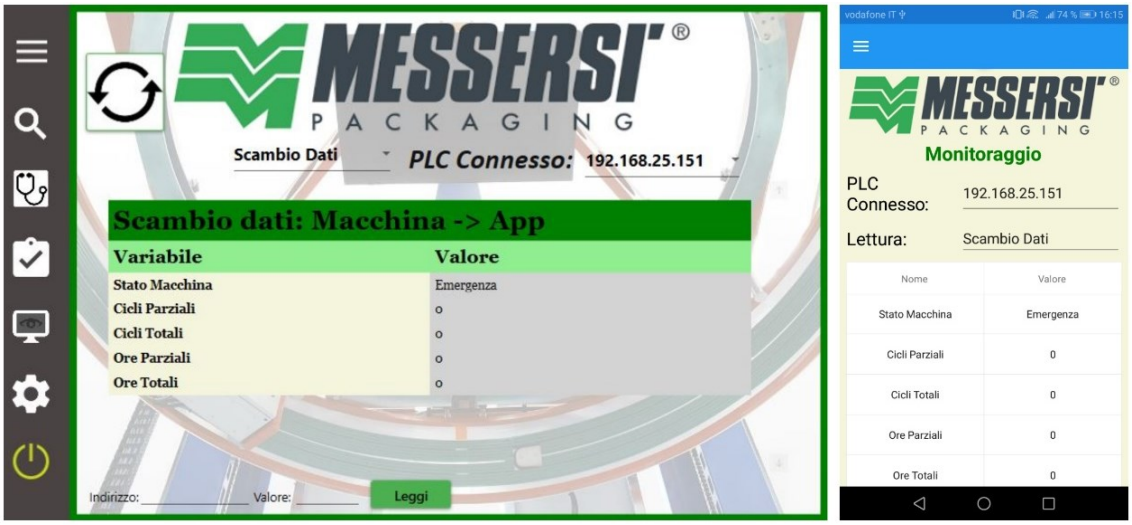

Figura 6.15 - Pagina di monitoraggio

- LetturaAllarmi(): preleva gli indirizzi degli allarmi dal database e in base al modello macchina ne esegue la lettura. Se uno o più allarmi sono attivi, il metodo MostraAllarme() ne ricava il testo sempre dal database e lo mostra in tabella.
- InvioNotifica_Click(): in caso di rilevamento di allarmi viene abilitato il pulsante per la notifica di segnalazione. Se ne viene richiesto l'invio, si procede all'invio dell'e-mail tramite il metodo InvioEmail().

Figura 6.16 - Notifica allarme

Il messaggio inviato conterrà i testi degli allarmi rilevati sulla macchina connessa:

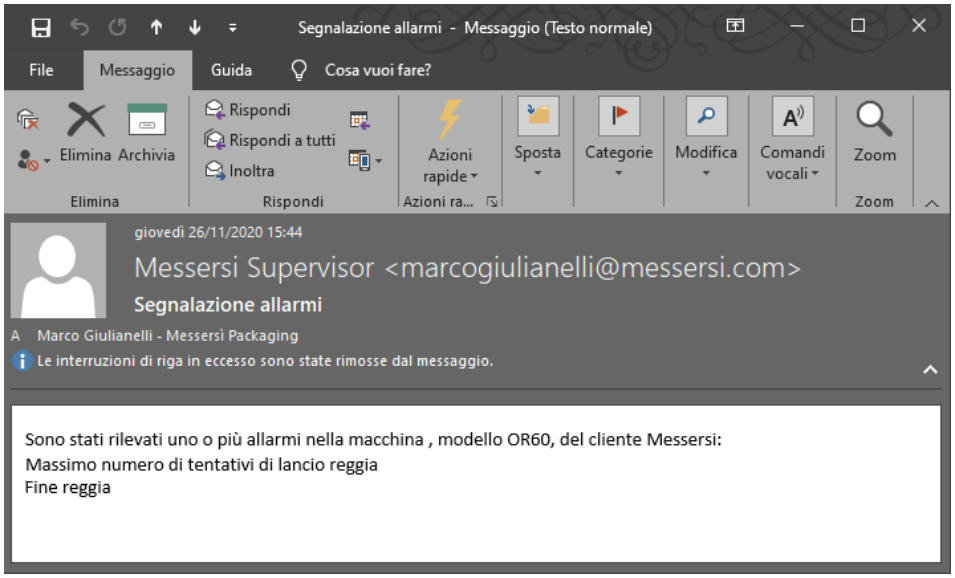

Figura 6.17 – E-mail di segnalazione

6.7 Impostazioni

Infine, nella pagina impostazioni è possibile modificare le informazioni di utenti e macchine salvate nel database MessersiApp.

Figura 6.18 - Pagina di impostazioni

In questa pagina viene ampiamente integrata la comunicazione con il database. Questa è infatti l'unica classe in grado di eseguire query di manipolazione.

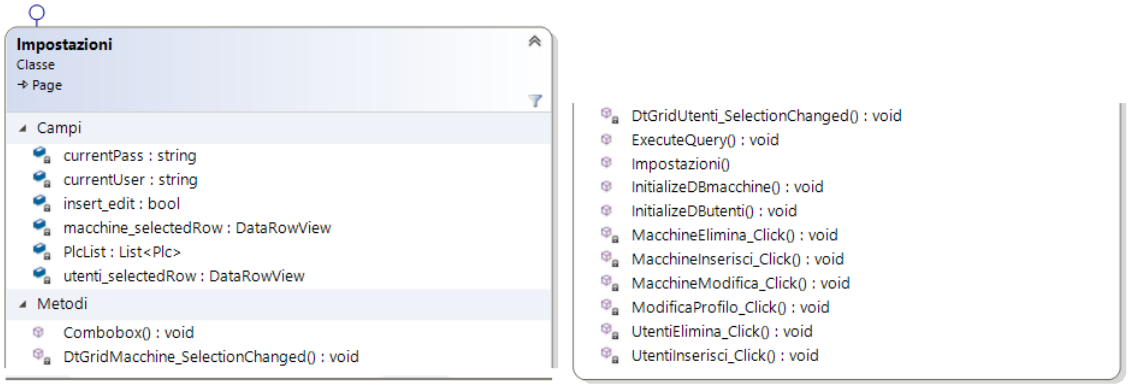

Figura 6.19 - Classe Impostazioni

- ExecuteQuery(): esegue la query fornita in base tipologia di modifica del database.
- currentUser e currentPass: contengono le credenziali dell'utente loggato.
- utenti_selectedRow: raccoglie i dati dell'utente selezionato nella apposita tabella.
- UtentiInserisci_Click(): apre il form di figura 6.20 per la registrazione di un nuovo profilo.
- UtentiElimina_Click(): elimina l'utente selezionato dal database.

Figura 6.20 - Inserimento nuovo utente

Analogamente, possono essere aggiunte, modificate ed eliminate macchine disponibili alla connessione. Per ogni macchina è obbligatorio indicare modello, marca e CPU del PLC, così da permettere una corretta gestione delle informazioni e garantire la funzionalità del supervisore.

- macchine_selectedRow: raccoglie i dati della macchina selezionata in tabella
- MacchineInserisci_Click(): apre il form di figura 6.21 per registrare una nuova macchina.
- combobox(): fornisce, i possibili modelli, brand e CPU selezionabili per l'inserimento di una nuova macchina.
- MacchineModficia_Click(): tramite il medesimo form permette la modifica dei dati delle macchine registrate.
- MacchineElimina_Click(): rimuove la macchina selezionata dal database. Non è possibile eliminare una macchina attualmente connessa.

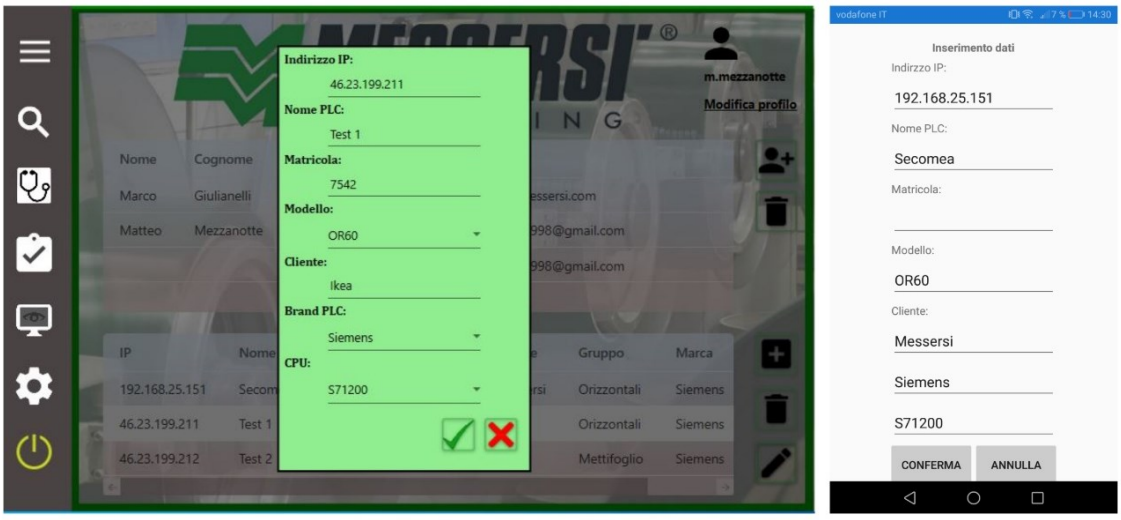

Figura 6.21 - Inserimento nuova macchina

Capitolo 7
Test e risultati

Durante e dopo lo sviluppo del codice sono state adoperate diverse modalità di test dell'applicativo. Viene quindi fornita una panoramica su obiettivi raggiunti e possibili future migliorie.

7.1 Test di collegamento

Lo sviluppo è stato condotto testando i progressi su PLC Siemens S7-1200 messi a disposizione dalla Messersì Packaging.

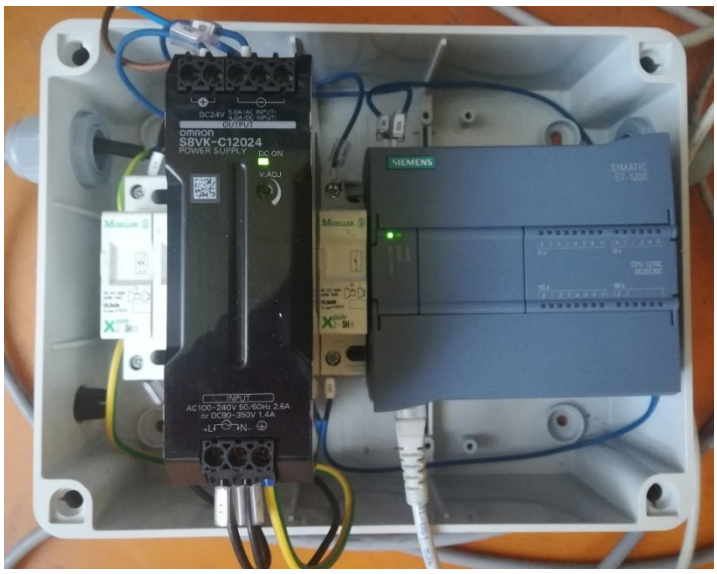

Figura 7.1 – PLC impiegato nei test

Su di essi sono stati caricati programmi standard per reggiatrici orizzontali, in modo da simulare uno scambio dati con una macchina reale. Il funzionamento dell'app è stato poi testato con diverse metodologie di connessione.

7.1.1 Connessione LAN

Inizialmente è si è testato il codice connettendosi a PLC in rete locale, ovvero collegati tramite porta Ethernet.

È stato sufficiente assegnare l'indirizzo IP al PLC tramite TIA Portal dalla scheda Indirizzi Ethernet, nel pannello di configurazione hardware del dispositivo.

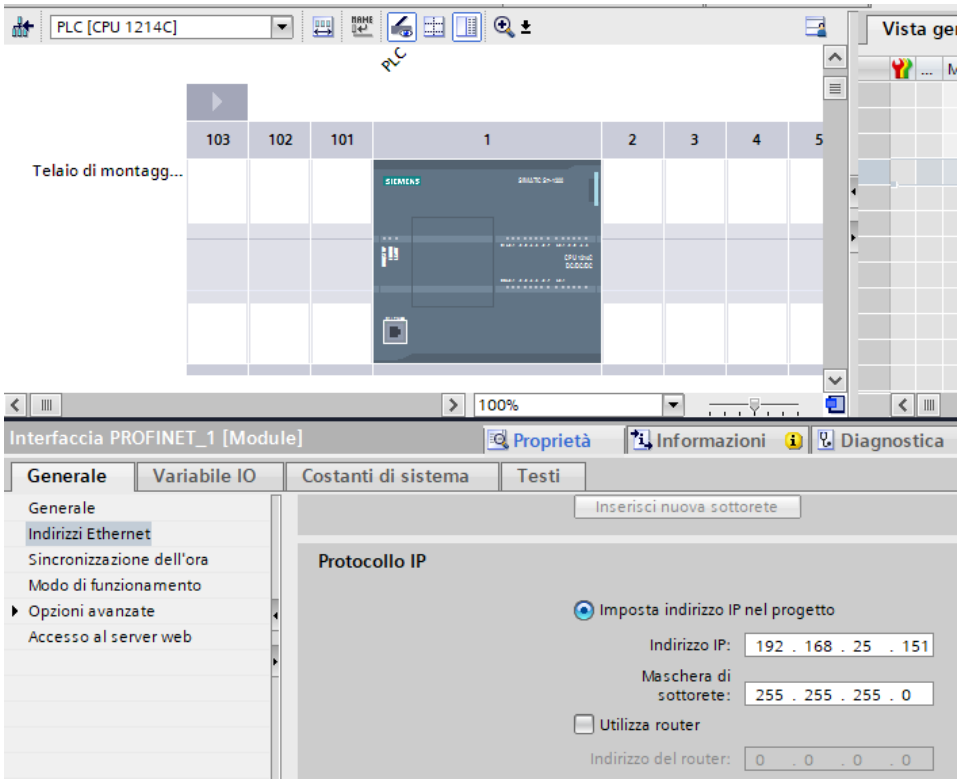

Figura 7.26 - Assegnazione indirizzo IP del PLC

7.1.2 Connessione da remoto

Per la connessione da remoto è stato invece necessario un procedimento diverso: l'amministratore di rete aziendale ha predisposto indirizzi IP pubblici ai quali sono stati associati altrettanti indirizzi privati interni alla rete. Si è quindi riusciti ad operare su uno o più PLC tramite connessione Internet da remoto.

Nella fattispecie, agli indirizzi privati 192.168.25.150 e 192.168.25.151 sono stati abbinati rispettivamente gli IP pubblici 46.23.199.211 e 46.23.199.21.

Pertanto, dal TIA Portal si è caricato l'indirizzo privato e un indirizzo di router, ovvero il gateway che permette al PLC di connettersi ad Internet. Una volta connesso alla rete Ethernet aziendale, l'IP privato viene associato al corrispondente indirizzo pubblico, raggiungibile per la connessione da remoto.

In figura 7.3 è mostrata la configurazione IP per la connessione da remoto sull'indirizzo 46.23.199.212.

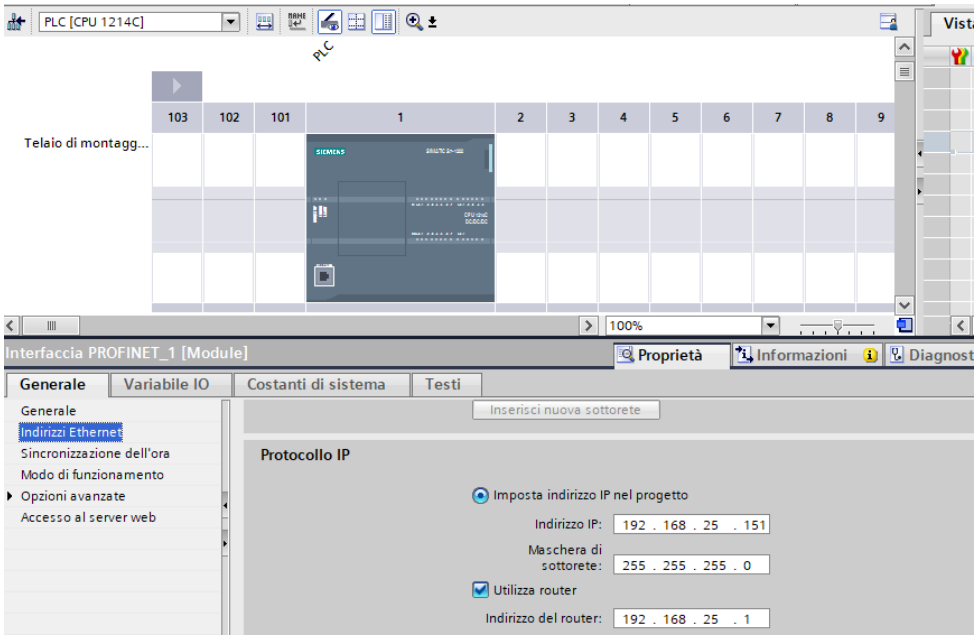

Figura 7.3 - Configurazione IP per connessione da remoto

Questa tecnica è però fortemente sconsigliata, in quanto connettere PLC di impianti industriali all'Internet pubblico comporterebbe gravi rischi per la sicurezza. Andrebbero piuttosto utilizzate reti VPN o dispositivi gateway con funzioni di protezione di rete.

7.1.2 Gateway Secomea

La connessione in sicurezza dei dispostivi industriali ad Internet è un aspetto cruciale per qualsiasi soluzione IoT.

Negli impianti che si doteranno di Messersì Supervisor l'accesso da remoto avverrà tramite appositi gateway quali il Secomea Site Manager, adoperato durante questa fase di test.

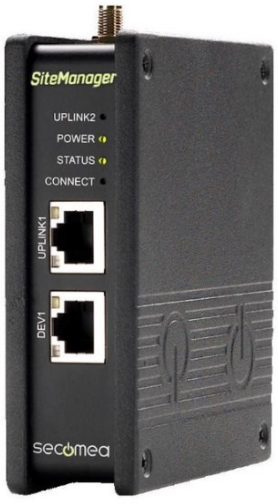

Figura 7.4 - Secomea Site Manager, directindustry.it

Si tratta di un dispositivo per soluzioni IoT che permette l'accesso da remoto e la raccolta dati dei dispositivi ad esso collegati, con la possibilità di settare un gran numero di opzioni hardware e software per le connessioni[13].

All'installazione, il Site Manager viene connesso a Internet tramite la porta Ethernet denominata UPLINK, mentre i dispositivi vengono connessi sulla porta DEV1. Le due porte sono però separate da un firewall che nega l'accesso ai dispositivi a utenti non autorizzati.

Tutte le configurazioni vengono eseguite sulla pagina web di Site Manager, alla quale può accedere solo il personale autorizzato avente le credenziali di accesso.

Per l'esecuzione dei test la porta UPLINK è stata quindi connessa alla rete utilizzando gli stessi indirizzi usati nel test precedente, mentre i PLC sono stati collegati alla porta DEV tramite switch Ethernet.

Figura 7.7 - Connessione PLC - Secomea

Dalla pagina web di Site Manager si accede al Gate Manager Agents, ovvero la visualizzazione dei dispositivi connessi al Secomea, dove vanno impostati nome, marca, tipologia e indirizzo IP dei PLC connessi alla porta DEV.

[13] https://www.secomea.com/sitemanager-industrial-iot-gateway/

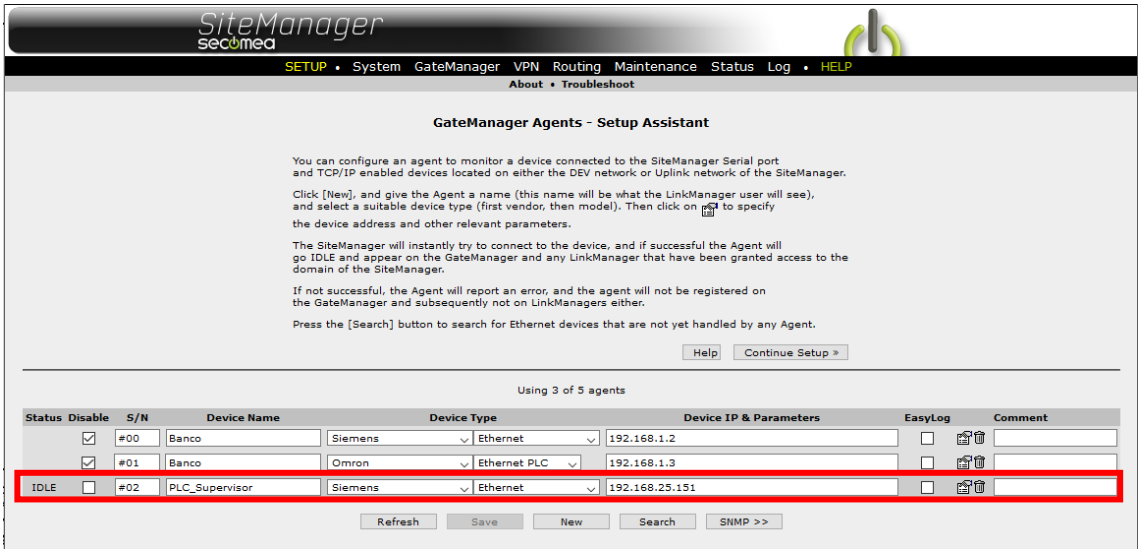

Figura 7.6 - Gate Manager Agents

Selezionando il PLC di interesse, si possono monitorare informazioni riguardanti lo stato della connessione con il dispositivo.

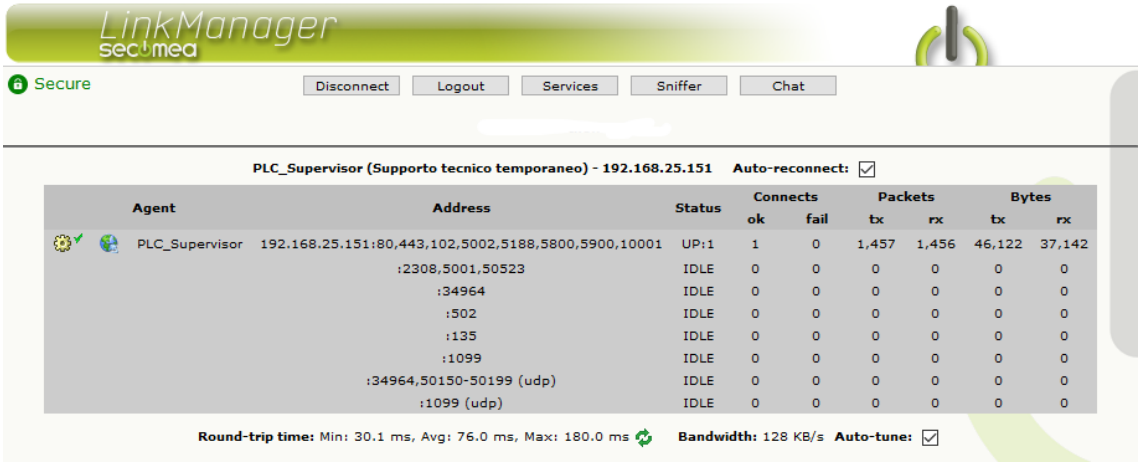

Figura 7.7 - Stato della connessione con PLC di test

Una volta configurata la rete, l'applicazione è stata in grado di connettersi ai PLC connessi sulla porta DEV, eseguendo tutte le funzionalità previste.

7.2 Problemi riscontrati

La maggior parte delle problematiche del progetto si è riscontrata nell'integrazione delle connessioni app-PLC e app-database. Il sistema necessita infatti di essere connesso al database MessersiApp per poter accedere alle informazioni richieste dalle principali funzioni. Allo stesso tempo va mantenuta la connessione con i PLC (o con i gateway d'impianto).

Proprio per questo motivo, è risultato impraticabile il proposito di connettere l'applicazione ai PLC tramite VPN, poiché una volta stabilita la connessione con il server della rete privata, il sistema operativo nega qualsiasi altra comunicazioni su reti diverse dalla VPN stessa.
Le altre interfacce di rete vengono disabilitate per questioni di sicurezza, impossibilitando la connessione al database.

7.3 Analisi prestazionale

Tramite gli strumenti di profilatura e diagnostica di Microsoft Visual Studio è stato possibile misurare le prestazioni del codice scritto, analizzando l'utilizzo della memoria e della CPU durante l'esecuzione dell'applicazione.
A tale scopo sono state eseguite due diagnostiche, analizzando un ciclo di funzionamento standard dell'applicazione WPF. Sono state eseguite in sequenza le seguenti azioni: login, connessione al PLC, check del collegamento, scrittura della DB100, lettura dello scambio dati, allarmi, ingressi/uscite e modifica del database utenti. Successivamente si è aperta una seconda connessione e si è ripetuta la procedura sul secondo PLC, per poi effettuare il logout.
La prima analisi ha riguardato l'allocazione della memoria usata dalla app: è possibile visualizzare le parti di codice in cui vengono allocati più oggetti, ovvero che occupano più memoria.

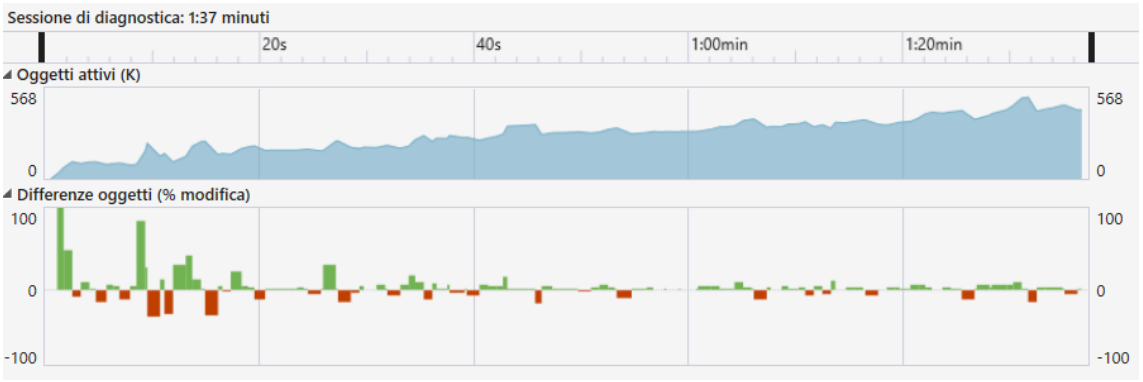

Figura 7.8 - Grafico di allocazione degli oggetti

Nel grafico superiore sono mostrati il numero di oggetti attivi, mentre in quello inferiore è illustrata la loro percentuale di modifica. Le parti verdi indicano quindi le nuove allocazioni, mentre quelle rosse rappresentano gli oggetti eliminati dalla *garbage collection*, tramite la quale il *Common Language Runtime* libera le porzioni di memoria non più utilizzate.
È quindi possibile risalire alle porzioni di codice più onerose in termini computazionali, che nella fattispecie risultano essere le inizializzazioni del markup e le chiamate delle funzioni di monitoraggio. Allo stesso tempo però, la *garbage collection* si dimostra in grado di mantenere l'utilizzo della memoria stabile, permettendo una buona fluidità d'esecuzione.

Il secondo test ha invece riguardato l'utilizzo della CPU da parte della app. A differenza dell'analisi precedente, questa analisi è influenzata dalle caratteristiche della macchina su cui viene eseguita. Per la prova è stato utilizzato un computer con specifiche di media fascia, quali processore AMD Ryzen 5 e memoria RAM da 6 GB.

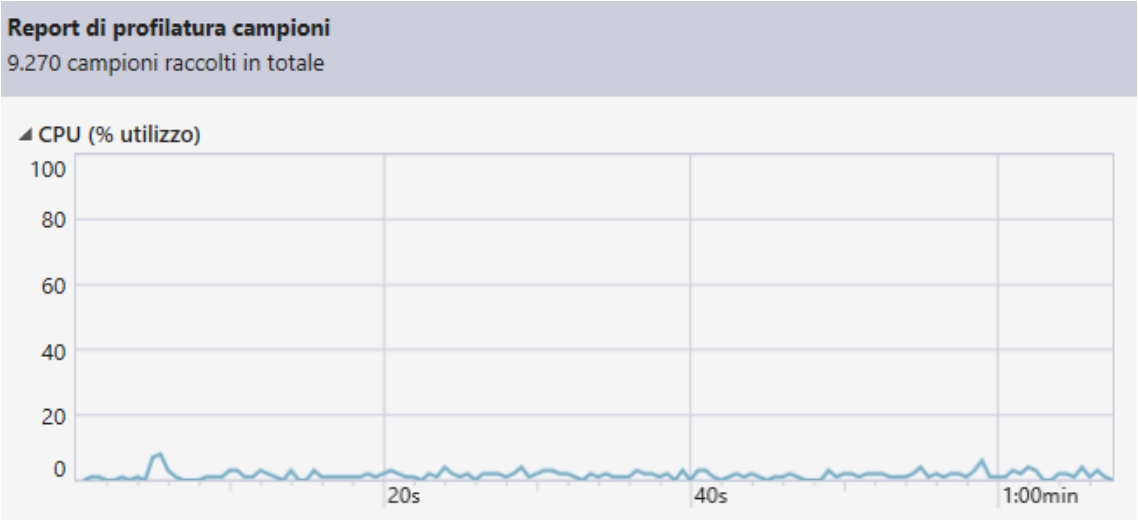

Figura 7.9 - Grafico di utilizzo della CPU

La sequenza temporale ha confermato le affermazioni fatte per l'allocazione della memoria, in quanto l'applicazione non ha sovraccaricato la CPU mantenendo una bassa percentuale di utilizzo per tutta la durata dell'esecuzione.

7.4 Obiettivi raggiunti

Nel corso di questi test è stato quindi appurato il funzionamento di Messersì la connessione a PLC remoti. Il test con Secomea ha anche confermato la possibilità di utilizzare l'app per connettersi a impianti ovunque locati.

Le funzioni di connessione, controllo e monitoraggio sono state testate con successo sia per la versione desktop che per quella mobile.

Il periodo di sviluppo si è concluso il 26/11/2020, con risultati soddisfacenti per l'azienda che ha espresso una valutazione positiva dei progressi raggiunti, ritenendo il sistema Messersì Supervisor un progetto da implementare negli impianti di produzione già esistenti e futuri.

Per quanto riguarda le specifiche inizialmente richieste, la maggior parte di esse sono state portate a termine. Altre sono state rimosse o rimandate poiché richiedevano un tempo di sviluppo maggiore di quello a disposizione.

I risultati raggiunti sono riassunti in tabella 7.1:

Punto	Descrizione	Stato
1	Definizione tabella generale per lo scambio dati per famiglia di macchine/impianti	Completato
2	Definizione struttura dell'applicazione da sviluppare su PC: home page e pagine secondarie di monitoraggio macchina, funzioni da implementare. Funzioni minime: 1. Pagina generale di verifica dei collegamenti macchine attivi 2. Pagina di diagnostica del collegamento (frame con messaggi di allarme o problemi di connessione) 3. Pagina di monitoraggio della macchina/impianto (visualizzazione della tabella dei dati definita + eventuali ulteriori funzioni)	Completato
3	Test del collegamento PC ↔ PLC Siemens S7-1200 in ufficio (rete locale)	Completato
4	Test del collegamento PC ↔ PLC Siemens S7-1200 in ufficio con ponte hotspot cellulare (da remoto)	Completato
5	Invio notifica (ad es. e-mail) in caso di valori anomali o presenza di allarmi sulla macchina	Completato
6	Test di connessione con 2 PLC Siemens S7-1200 attivi	Completato
7	Test di connessione a PLC Siemens S7-1200 via dispositivo di connessione remota (Secomea)	Completato
8	Test di connessione a PLC Siemens S7-1200 via VPN	Annullato
9	Implementazione su PLC Siemens S7-1200 di un sistema di invio notifica alla Messersì per presenza allarmi o necessità di manutenzione	Rimandato
10	Test del sistema con PLC di brand diversi da Siemens	Rimandato

Tabella 7.4 - Obiettivi raggiunti

7.5 Possibili migliorie

Il progetto allo stato attuale risulta in grado di eseguire operazioni di controllo e monitoraggio, ma in futuro potrebbe ampliarsi implementando nuove funzioni avanzate.

La prima miglioria da apportare sarebbe quella di estenderne l'uso a PLC non di marca Siemens. Sono infatti disponibili librerie open source analoghe a S7.Net, ad esempio per PLC Omron

(mcOmron) o Allen Bradley (LibPlcTag), entrambe funzionanti tramite protocollo IP e scritte in codice C#, facenti parte di .NET e perciò facilmente integrabili.
La seconda funzione da implementare è quella di aggiungere un sistema di assistenza che connetta direttamente il cliente con il servizio Messersì in caso di malfunzionamenti. Si prevede di aggiungere una apposita pagina assistenza sia sulla app desktop che su quella mobile. Da quest'ultima il cliente potrà scannerizzare codici QR posti a bordo macchina per visualizzare manuali o guide per la risoluzione di malfunzionamenti. In alternativa, potrà inviare un messaggio di segnalazione al servizio assistenza Messersì, che lo visualizzerà nella corrispondente pagina nell'app desktop.
Contestualmente a questo servizio ci si appoggerà a una piattaforma cloud per l'invio di notifiche push sui dispositivi, in modo da realizzare un sistema di teleassistenza attivo in ogni istante.
Altre possibili soluzioni potrebbero nascere durante l'esecuzione di test in officina durante i collaudi e, soprattutto, dall'esperienza d'uso presso il cliente.

Capitolo 8
Conclusioni

Il progetto descritto in questa trattazione ha cercato di fornire un sistema valido per il monitoraggio e lo scambio dati con PLC installati su macchinari industriali, in questo caso nel settore del packaging.

A tal fine, i due applicativi sviluppati si sono dimostrati in grado di permettere lo scambio di informazioni richiesto sia da dispositivi fissi che mobile.

Le applicazioni basano il loro funzionamento su piattaforme software largamente diffuse e sono quindi installabili su quasi la totalità dei dispositivi in commercio.

Si è quindi realizzato un software in grado di svolgere operazioni di controllo e monitoraggio su PLC connessi ad Internet in rete locale ma soprattutto da remoto. In aggiunta è stato implementato un database di raccolta dati che potrà fungere da archivio macchine per l'azienda. Il risultato ottenuto soddisfa i requisiti richiesti nelle fasi iniziali del progetto, permettendo l'interconnessione dei macchinari Messersì tramite un'unica piattaforma online.

Il risultato ottenuto da tale lavoro rispecchia il quadro delle innovazioni portato dalla quarta rivoluzione industriale. Grazie anche a strumenti tipici dell'Internet of Things, una notevole quantità di informazioni può essere ora ricavata dagli impianti produttivi, per poi essere sottoposta a viarie elaborazioni a seconda delle necessità.

Ringraziamenti

Il lavoro trattato in questo elaborato è stato svolto durante i mesi di ottobre e novembre 2020 presso la sede di Barbara (AN) della Messersì Packaging. Un ringraziamento va quindi a tutto il team dell'azienda che mi ha accolto e permesso di lavorare a questo progetto.

La trattazione è stata scritta sotto la supervisione del professor Giuseppe Conte, al quale sono grato per la sua disponibilità e il suo supporto.

Bibliografia

[1] Schwab K., *La quarta rivoluzione industriale*, Milano, Franco Angeli, 2019

[2] Ministero dello Sviluppo Economico, *Piano Industria 4.0*, 2017

[3] https://www.internet4things.it

[4] Giuliano Ortolani, Ezio Venturi, *Manuale di Elettrotecnica, Elettronica e Automazione,* Milano, Hoepli, 2017

[5] Capital Normal University, *Component-based Formal Modeling of PLC Systems,* Beijing, Hindawi Publishing Corporation, 2013

[6] Charles Petzold, *Creating Mobile Apps with Xamarin.Forms*, Redmond, Washington, Microsoft Press, 2016

[7] https://docs.microsoft.com/

[8] Emanuele Frontoni, Adriano Mancini, *Programmazione ad oggetti per l'Ingegneria Informatica,* McGraw-Hill, 2019

[9] https://www.wpf-tutorial.com/

[10] Paolo Camagni, Riccardo Nikolassy, *Database SQL & PHP*, Milano, Hoepli, 2018