UNIVERSITÀ
POLITECNICA
DELLE MARCHE

Facoltà di Ingegneria
Corso di Laurea in Ingegneria Informatica e dell'Automazione

Acquisizione dati da sistemi di automazione per monitoraggio delle prestazioni di macchinari e di operatori umani su una linea di produzione automatizzata

Data acquisition from automation systems to monitor the performance of machinery and human operators on an automated production line

Candidato:
Davide Balducci

Relatore:
Prof. Andrea Bonci

Correlatore:
Prof. Mariorosario Prist

Anno Accademico 2022-2023

UNIVERSITÀ
POLITECNICA
DELLE MARCHE

Facoltà di Ingegneria
Corso di Laurea in Ingegneria Informatica e dell'Automazione

Acquisizione dati da sistemi di automazione per monitoraggio delle prestazioni di macchinari e di operatori umani su una linea di produzione automatizzata

Data acquisition from automation systems to monitor the performance of machinery and human operators on an automated production line

Candidato:
Davide Balducci

Relatore:
Prof. Andrea Bonci

Correlatore:
Prof. Mariorosario Prist

Anno Accademico 2022-2023

Università Politecnica delle Marche
Facoltà di Ingegneria
Corso di Laurea in Ingegneria Informatica e dell'Automazione
Via Brecce Bianche – 60131 Ancona (AN), Italy

Ai miei genitori che mi hanno sempre sostenuto,
ai miei amici che sono come una famiglia.

Ringraziamenti

Innanzitutto vorrei ringraziare i Professori Andrea Bonci e Mario Prist per avermi aiutato lungo il periodo di tirocinio e nella stesura della tesi.

Le prime persone a cui dedico questo primo traguardo sono i miei genitori, Felice e Michela, i quali mi hanno permesso di svolgere questo lungo e tormentato percorso sostenendomi sempre in ogni momento, grazie davvero.
Ringrazio con tutto il mio cuore i miei coinquilini: Flavio, Emanuele, Richard, Luigi, Andrea, Francesco e Filippo, senza di voi la mia vita sarebbe un po' più grigia.
Ringrazio Marco, il mio primo coinquilino e vecchio compagno di corso, è anche grazie a te se sono riuscito ad arrivare dove sono ora, non sai quanto mi dispiace che le nostre strade si siano separate.
Un ringraziamento speciale è rivolto a Xhesika e Beatrice, la vostra amicizia è un valore aggiunto di enorme importanza per me, spero di non perdervi in futuro.
Ringrazio i compagni di corso conosciuti sin dai primi giorni di università: Andrea, Carlo, Arianna, Michele, Alessandro e Marco, per aver alleggerito quelle infinite ore di lezione passate sui banchi. Un grazie particolare va ad Andrea, con il quale ho preparato gran parte degli esami tra un "non ce la faccio" e "questo lo do al prossimo appello", alla fine ci siamo riusciti anche noi amico mio.
Ringrazio tutti i ragazzi del gruppo di Atletica con il quale ho legato al primo anno, conoscere voi e avere un gruppo unito con cui allenarsi e divertirsi in un momento di cambiamenti, come il trasferirsi in una nuova città, mi è stato di grande aiuto. Devo ancora sdebitarmi per gli infiniti passaggi che mi avete dato.
Ringrazio infine tutte le fantastiche persone che ho conosciuto in questi anni con cui ho condiviso bellissime esperienze e che porterò sempre con me.

Ancona, Luglio 2023

Davide Balducci

Sommario

L'elaborato tratta lo sviluppo software di un metodo per l'acquisizione dati da sistemi di automazione per il monitoraggio delle prestazioni di macchinari e operatori umani su una linea di produzione. Partendo dalla descrizione della linea automatizzata CP-Factory (Cyber-Physical Factory) presa in esame per lo sviluppo della tesi, per poi passare ad una introduzione al software MES4 (Manufacturing Execution System) utilizzato per comunicare con la linea di produzione ed una descrizione dettagliata del database consultato per l'ottenimento dei dati necessari, vengono fornite tutte le informazioni utili per la comprensione del lavoro svolto. Successivamente viene esposta la teoria associata agli indici di prestazione utilizzati con le rispettive formule, è inoltre riportata una mappatura dei parametri prelevati dal database e associati alle variabili degli indicatori chiave di prestazione. In seguito vengono riportate le prove sperimentali con lo scopo di confermare la validità degli indicatori proposti e del software sviluppato ai fini del loro monitoraggio, per poi giungere ad un'analisi accurata del software suddiviso nelle sue principali fasi. Infine si giunge all'estensione dell'indicatore chiave di prestazione a livello di fabbrica attraverso un'adeguata interconnessione tra sottosistemi, per poi analizzare i risultati ottenuti su una produzione consistente nella prova finale. Infine è poi stata sviluppata una interfaccia grafica per mostrare attraverso i grafici la condizione operativa della linea di produzione automatizzata.

Indice

Elenco delle figure

Elenco delle tabelle

Capitolo 1.

Cyber Physical Factory

Le attrezzature dedicate ai sistemi di produzione Cyber Fisici, come la linea di produzione automatizzata Cyber Physical Factory (CPFactory, Figura 1.1) prodotta da FESTO C.T.E. SRL, attualmente dislocata presso il laboratorio i-Labs di Jesi e accessibile tramite il laboratorio di automazione del Dipartimento di Ingegneria dell'Informazione (DII) della Facoltà di Ingegneria, sono destinate a creare un ambiente produttivo analogo ad una linea di produzione reale nella quale siano implementate alcune delle più promettenti tecnologie per l'automazione della produzione, l'integrazione e l'automazione dei processi e la tracciabilità di prodotto e/o processo.

Figura 1.1.: CPFactory.

Il design modulare e flessibile permette agli utilizzatori di modellizzare e lavorare in diversi scenari: dal semplice sistema di trasferimento pallet con controllo integrato, alla struttura di produzione in rete con servizi cloud.

1.1. Descrizione della CPFactory e del ciclo di produzione

Il ciclo standard implementato è tipico di molti processi industriali discreti ed è realizzato mediante 4 stazioni CPFactory (Figura 1.2).

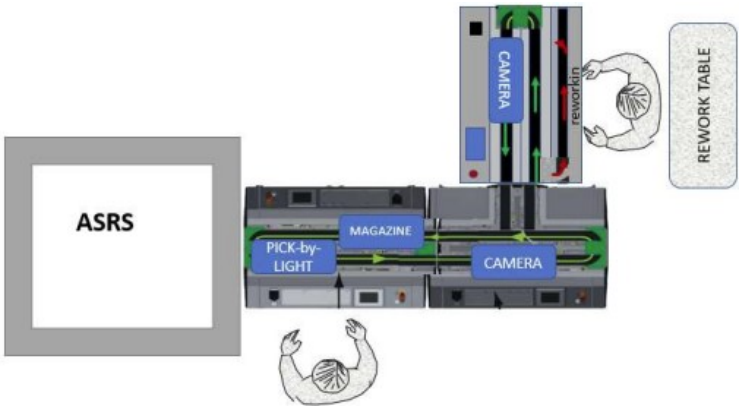


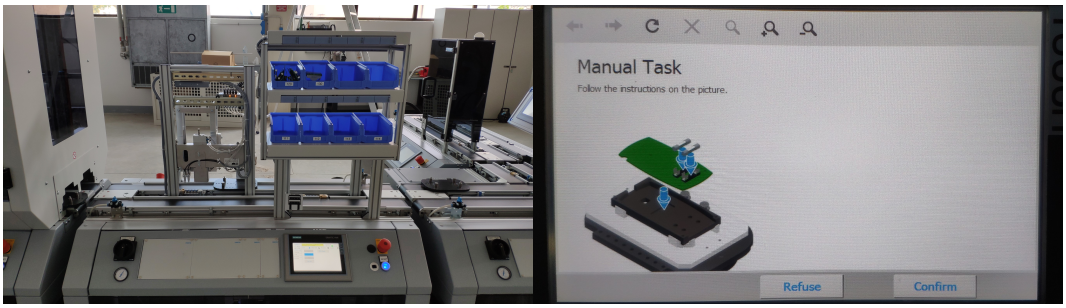
Figura 1.2.: Ciclo di produzione.

Il ciclo automatizzato inizia dopo aver caricato manualmente il magazzino AS/RS con i gusci posteriori sopra al pallet, è possibile caricare il magazzino da 1 a 32 pezzi, ma a causa di una disponibilità limitata dei pallet in laboratorio il numero di pezzi massimo è stato ridotto a 8.

1. Inizio del processo: al lancio di un determinato ordine il sistema AS/RS (Figura 1.3) immette sul ciclo un pallet completo di guscio posteriore, e lo identifica univocamente tramite codice RFiD, che rimarrà tracciato per tutto il processo all'interno del MES.

Figura 1.3.: AS/RS.

2. Nella prima postazione l'application module Pick by Light ha il compito di guidare l'operatore, con l'aiuto di indicatori luminosi, sulla corretta sequenza di assemblaggio della scheda e del guscio (Figura 1.4).
   L'operatore inoltre può intenzionalmente o casualmente introdurre degli errori di montaggio, che vengono rilevati nella stazione successiva.



(a) Pick by Light. (b) Immagine su HMI.

Figura 1.4.: Pick by Light.

3. Sulla seconda postazione, mediante fotocamera intelligente (Figura 1.5), si vanno a verificare diversi parametri del prodotto parzialmente assemblato (presenza/assenza PCB, presenza/assenza fusibili, posizione corretta, colore corretto, ecc). In base all'esito di difettosità il prodotto (corretto) prosegue verso la terza postazione, sul ciclo standard (conveyor rettilineo); oppure il prodotto (difettoso) viene deviato verso la rilavorazione (branch), che avviene nella stazione innestata a T sul ciclo principale.

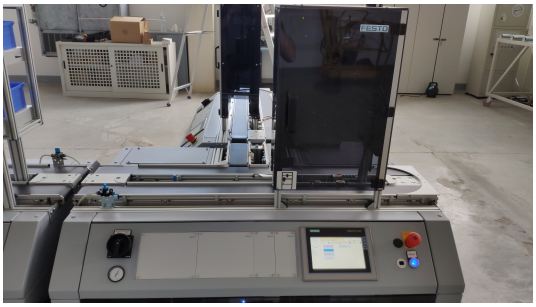
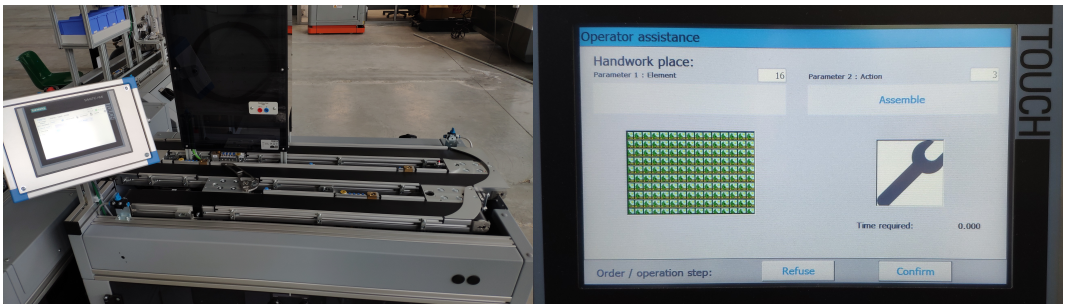


Figura 1.5.: Fotocamera 1 per ispezione qualità.

4. Descrizione delle lavorazioni alternative:
   a) Se il pezzo risulta correttamente assemblato, esso viene avviato verso la terza postazione sul circuito principale, rappresentato dal magazzino dei coperchi (Figura 1.6), di tipo a gravità e gestito a logica FIFO, i coperchi vengono deposti sul pezzo presente nel pallet, a finire il prodotto stesso.

Figura 1.6.: Magazzino distributore.

b) Se il pezzo è difettoso, esso viene avviato verso la quarta postazione che si trova sul ramo secondario del circuito, detta stazione di rilavorazione (rework, Figura 1.7), costituita da tre nastri; due di essi formano un circuito principale sul quale viene eseguita la correzione dell'errore da un operatore umano.



(a) Stazione di Rework. (b) Pannello HMI.

Figura 1.7.: Rework.

5. Nella stazione di rilavorazione i pezzi riparati vengono sottoposti a nuovo test attraverso un'altra fotocamera intelligente (Figura 1.8). A valle di quest'ultima i pezzi vengono nuovamente immessi nel ciclo principale destinandoli al magazzino finale, o definitivamente marcati per lo scarto se si rilevano ulteriori difformità.

Figura 1.8.: Fotocamera 2 per ispezione qualità.

6. Fine del processo: nel magazzino AS/RS, sulla parete opposta a quella dei semilavorati, vengono immagazzinati tutti i pezzi prodotti sino a soddisfacimento dell'ordine impostato da MES; quelli contrassegnati come definitivamente difettosi al secondo test vengono accantonati in una zona apposita del magazzino stesso per successivo scarto.

1.2. Il prodotto campione

Il sistema nel suo insieme realizza l'assemblaggio di un componente elettronico a partire da 4 parti semilavorate comprendenti: coperchio, guscio posteriore, scheda elettronica PCB, una coppia di fusibili.
Il coperchio (guscio anteriore), di cui esistono diverse varianti di colore, completa il prodotto finito in uscita dall'impianto.
La scheda a circuito stampato può essere impacchettata tra i due contenitori, completata da un set di 1 o 2 fusibili, secondo l'ordine di produzione. Esistono varianti di colore del guscio.
Il guscio posteriore, anch'esso disponibile in più varianti di colore, accoglie la scheda PCB ed in seguito viene chiuso dal coperchio.

Il prodotto campione che viene man mano assemblato e/o lavorato, viene trasportato in un pallet, identificato da un tag RFiD associato univocamente al prodotto che trasporta. Tale tag viene letto all'ingresso di ogni modulo applicativo, e scritto in corrispondenza dell'uscita da esso dopo le operazioni subite (Figura 1.9).

(a) Carrier. (b) Pallet. (c) Tracciamento.

Figura 1.9.: Trasporto.

1.3. Descrizione delle unità funzionali della CPFactory

Da un punto di vista hardware, il sistema è costituito da un insieme di macchine che riproducono in ambiente controllato, ovvero appunto uno spazio laboratoriale, un reale impianto di produzione polifunzionale completo, nel rispetto dei canoni definiti dalla piattaforma Industry 4.0. Tale impianto possiede caratteristiche di modularità, flessibilità operativa e funzionale, espandibilità, ed è completamente aperto con l'obiettivo di poterlo integrare a livello funzionale con altre componenti

del laboratorio i-Labs.
Il concetto di *modularità* è riferito alla possibilità di poter utilizzare funzionalmente i singoli moduli costituenti l'impianto sia in configurazione stand alone, sia in rete con gli altri, con interfacce meccaniche, elettriche e protocolli standard.
Il concetto di *espandibilità* è riferito al fatto che si possa estendere la funzionalità dell'impianto mediante l'aggiunta di ulteriori moduli del fornitore.
Il concetto di *flessibilità* operativa è previsto come insito nei diversi moduli ed è in linea con quanto definito dalla piattaforma I4.0 come macchina "Plug&Produce".

L'impianto può operativamente svolgere un ciclo di produzione di artefatti reali formati da più componenti, forniti in più punti dell'impianto stesso mediante assemblaggio con operatore umano o robotico. Ne discende la possibilità di poter avere molteplici varianti di prodotto finito.

Nel rispetto della modularità funzionale, ciascun modulo è costituito da una base comune contenente un organo di movimentazione con nastro trasportatore (conveyor-pallet transfert system), bidirezionale a doppia e/o a tripla linea; ciascuno autonomamente programmabili tramite apposito Controller con interfacce di rete standard (tipo Profinet, Ethernet, OPC-UA, ecc.). La veicolazione del prodotto da assemblare/lavorare avviene tramite appositi pallet trasportati da carrier intelligenti, ovvero dotati di un sistema di tracciabilità basato su tecnica RFiD.

Il conveyor-pallet transfert system possiede un quadro comandi HMI (Human-Machine Interface) con display touch, un sistema di blocco del pallet, un lettore/scrittore RFiD, ed un controllore modulare integrato in rete, in grado di poter comunicare con diversi protocolli standard prima indicati. Ciascun modulo avrà unità funzionali automatizzate, dotate di loro controllo autonomo che comunica con l'unità base stessa mediante l'utilizzo di scambio segnali I/O digitali ed analogici, IO-link o Profinet.

1.3.1. Moduli applicativi

Ciascun modulo è costituito da una base comune contenente un organo di movimentazione con nastro trasportatore (Conveyor) ed uno o più *Application Modules* (moduli applicativi) installati sopra al Conveyor per effettuare le lavorazioni sul prodotto. L'insieme delle trasformazioni che vengono operate sul prodotto lungo il percorso avvengono dunque in punti precisi del Conveyor, in corrispondenza dei moduli applicativi. Ogni segmento del Conveyor porta un solo modulo applicativo. Possono dunque essere effettuate nei differenti moduli applicativi azioni automatizzate, manuali e robotizzate.

Ciascun modulo applicativo è un apparato specializzato e configurabile, in grado di operare sui pezzi, ed è montato a cavallo di ciascun Conveyor. Poiché ogni stazione CPFactory è dotata di due (o tre) Conveyors, essa può montare fino a tre moduli applicativi.

Nel sistema CPFactory è il pezzo (anche chiamato prodotto) che guida l'evoluzione del processo imponendo:

- il routing verso il prossimo step, all'uscita di un modulo applicativo;
- la riconfigurazione del modulo applicativo al suo ingresso, per esempio un differente schema di illuminazione delle parti da assemblare nella stazione Pick By Light per differenti pezzi (ulteriore descrizione su Pick By Light a seguire);

in base alle caratteristiche associate al lotto cui il pezzo stesso appartiene.

In Figura 1.10 viene raffigurato lo schema a blocchi della CPFactory, dove per ogni Conveyor (in figura descritto con "module") sono installati uno o più moduli applicativi.

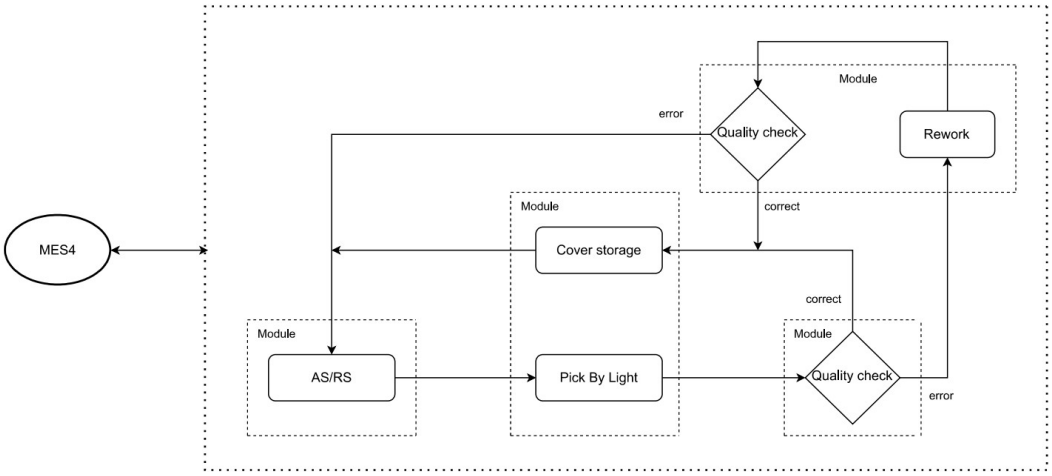


Figura 1.10.: Schema a blocchi della CPFactory.

I moduli applicativi presenti sulla CPFactory nei laboratori i-Labs sono i seguenti:

- Magazzino AS/RS (Automated Storage and Retrival System, Figura 1.3): questo modulo applicativo, attraverso una pinza montata su un braccio robotico a 3 assi, consente il posizionamento dei pezzi grezzi sul nastro trasportatore dal magazzino dei gusci posteriori, a ciclo ultimato effettua invece il prelevamento dei pezzi finiti (idonei o di scarto) dal nastro trasportatore al magazzino. In Figura 1.10 è rappresentato con il nome "AS/RS".
- Assemblaggio guidato (Pick by Light) della scheda elettronica (Figura 1.4a): in questo modulo applicativo il prodotto sul pallet rimane in attesa finché l'operatore umano non dà la conferma attraverso il pannello touch screen HMI che la produzione può continuare. Infatti, per l'assemblaggio dei pezzi sul prodotto, l'operatore può seguire le istruzioni sempre dal pannello HMI (Figura 1.4b) e prelevare, con l'ordine dettato dall'illuminazione, i pezzi da assemblare dagli appositi scompartimenti che si illuminano con un led verde. In Figura 1.10 è rappresentato con il nome "Pick By Light".

- Fotocamera di ispezione qualità (Figura 1.5 e Figura 1.8): queste due fotocamere in meno di un secondo analizzano il prodotto sottostante e decretano se il prodotto è stato assemblato dall'operatore umano correttamente o meno attraverso un indicatore luminoso posto sopra il modulo applicativo, verde se il prodotto è buono, rosso se è sbagliato. La fotocamera 1 analizza tutti i prodotti ed è posizionata subito dopo il modulo Pick by Light, mentre la fotocamera 2 è collocata dopo la stazione di rework ed è attiva dunque solo quando il prodotto ha appena subito una operazione di rework. In Figura 1.10 sono rappresentate con il nome di "Quality check".

- Magazzino distributore dei gusci anteriori (Figura 1.6): il prodotto sul pallet composto da guscio anteriore e scheda PCB, con o senza fusibili, viene completato da questo modulo applicativo; attraverso la combinazione di due pistoni questo magazzino posiziona sul prodotto un guscio anteriore. In Figura 1.10 è rappresentato con il nome di "Cover storage".

- Stazione di rilavorazione pezzo (Figura 1.7): anche in questo modulo applicativo il prodotto sul pallet rimane in attesa finché l'operatore umano non dà la conferma attraverso il pannello touch screen HMI che la produzione può continuare. Il prodotto può finire in questa stazione nel caso l'ispezione della prima fotocamera non abbia dato esiti positivi oppure per la pressatura manuale del guscio anteriore. In Figura 1.10 è rappresentato con il nome di "Rework".

La Figura 1.11 è stata inserita allo scopo di descrivere maggiormente il ciclo di produzione della CPFactory, già introdotto dallo schema a blocchi in Figura 1.10 utilizzato per spiegare ulteriormente i moduli applicativi appena citati. In questo diagramma dunque è presente il percorso che il prodotto effettua sulla CPFactory, dove sono state descritte maggiormente le situazioni in cui il pezzo è idoneo al controllo qualità delle Camere 1 e 2, oppure se viene classificato come non idoneo. Ogni blocco rappresenta il modulo applicativo coinvolto, sono poi indicati per ogni modulo i tempi di lavorazione e, per ogni spostamento da un modulo applicativo ad un altro, i tempi di trasporto.

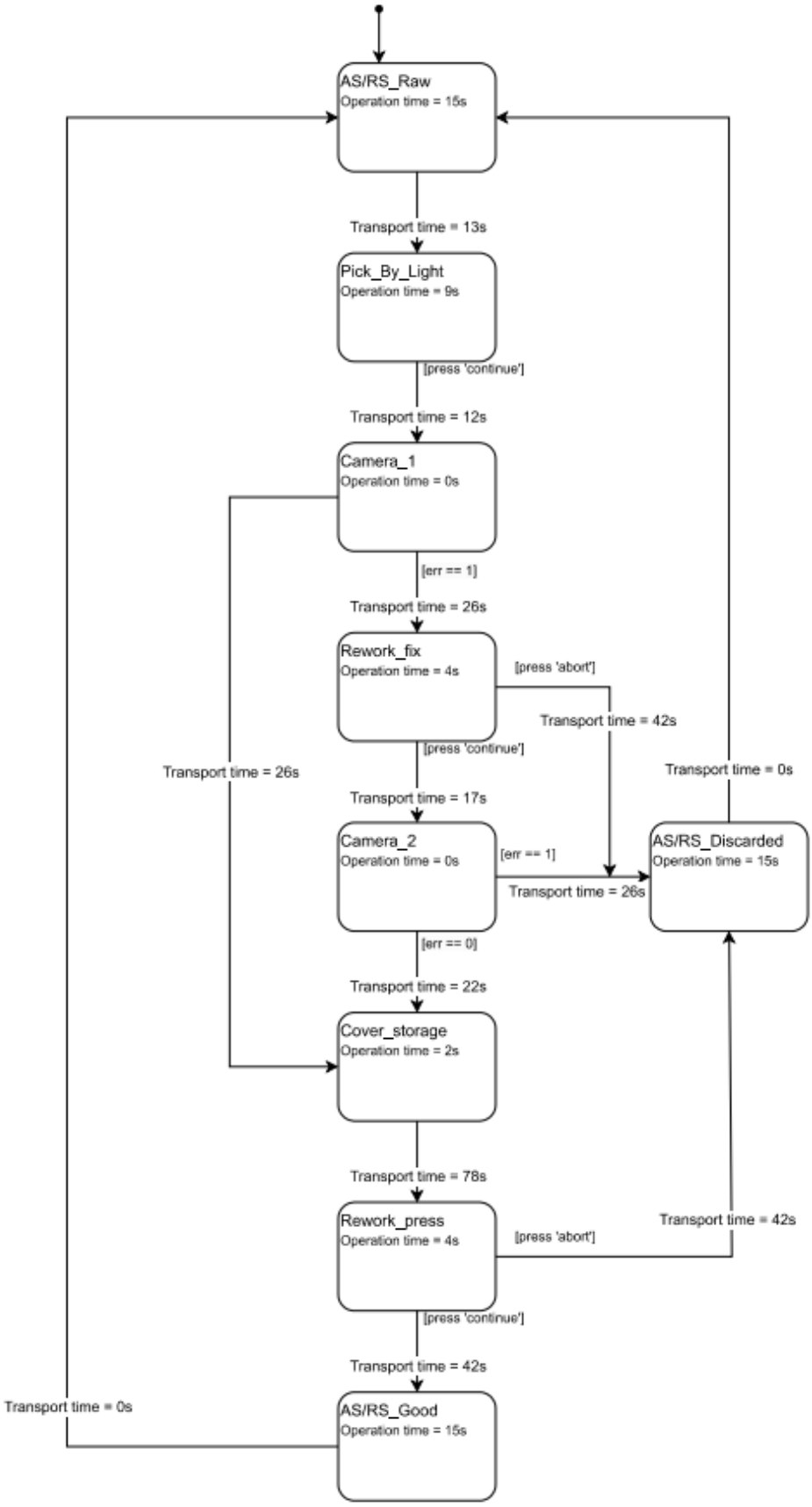


Figura 1.11.: Macchina a stati della CPFactory.

1.4. Procedura di accensione della macchina

Per l'accensione della CPFactory è consigliata la seguente procedura:

- Collegare all'alimentazione il compressore ad aria e attendere il suo riempimento. Vengono riportate le uniche informazioni riguardanti il compressore presenti nel manuale originale "alcuni moduli prevedono l'alimentazione ad aria compressa, con portata d'aria ridotta, generabile con un compressore elettrico di piccola taglia". Durante l'utilizzo della macchina la pressione impostata è stata di 6 bar e non oltre.

- Prima di accendere la CPFactory, è buona pratica soffiare con l'aria compressa su tutti i moduli della macchina, in particolare assicurarsi di avere pulito bene i sensori di posizione presenti agli estremi di ogni modulo da possibili residui di polvere che altrimenti potrebbero dare falsi segnali al sistema.

- Collegare l'aria compressa alla macchina e girare su ON l'interruttore principale posizionato sul modulo dell'AS/RS.

- Per ogni modulo a seguire: 1) girare su ON l'interruttore, 2) disattivare il pulsante di emergenza, 3) premere il pulsante di reset posizionato sotto al pulsante di emergenza, 4) cancellare gli errori passati sul pannello HMI integrato in ogni stazione (Figura 1.12, i numeri in rosso rispecchiano i passaggi da compiere nell'ordine descritto).

Figura 1.12.: Procedura accensione.

- Infine accendere il computer lì presente, collegare il cavo ethernet della macchina, accedere al computer (password: festo) e aprire il MES.

Capitolo 2.

MES4

2.1. Introduzione

MES4 è un *Manufacturing Execution System* appositamente concepito per essere applicato su piattaforme Industry 4.0.
MES4, una volta collegato in rete con tutti i PLC locali distribuiti nelle stazioni della CPFactory, permette di impostare gli ordini di produzione, configurare gli Application Module, assegnare ai Carrier l'instradamento tra moduli in base ai prodotti che essi trasportano e contemporaneamente di riconfigurare i moduli in tempo reale in base al prodotto da cui essi sono in quel momento attraversati.
In MES4 viene creata dagli operatori una catena di uno o più ordini indipendenti, per espletare i quali vengono determinate le sequenze delle operazioni avviate o completate in ogni stazione [1].

2.2. Finestra Principale

Nella parte superiore della finestra principale (Figura 2.1) si può trovare la barra del menu. Può essere usata per cambiare l'utente, aprire tools speciali e organizzare le sottofinestre.

Sul lato sinistro della finestra principale invece è presente il menu principale con i pulsanti designati ad aprire le sottofinestre. Questi pulsanti sono organizzati in 4 gruppi:

1. Production Control: finestra per il monitoraggio dello stato corrente della produzione.

2. Order Management: finestra per la gestione degli ordini, ovvero creazione di ordini, monitoraggio di ordini già creati o in corso.

3. Quality Management: finestra per osservare il report sull'efficienza delle diverse stazioni di lavoro e degli ordini eseguiti.

4. Master Data: finestra per aggiungere, configurare o eliminare parti, processi di lavoro e stazioni.

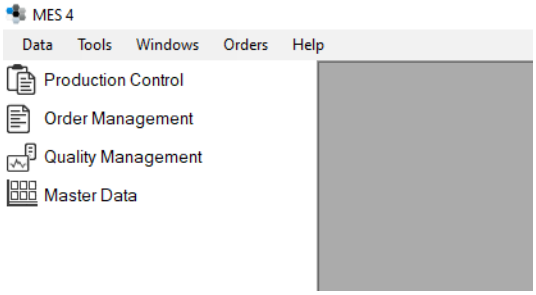


Figura 2.1.: Finestra Principale.

Questi gruppi hanno una funzione a cartella, si possono aprire tali gruppi cliccandoci sopra. A seguire verranno descritte più nel dettaglio le sezioni dei gruppi maggiormente utilizzati durante lo sviluppo dell'elaborato, per ulteriori informazioni è possibile consultare i manuali della CPFactory disponibili nel laboratorio i-Labs.

2.2.1. Production Control - Buffers

Nelle posizioni del buffer, i pezzi sono immagazzinati con un numero. Nella finestra Buffers (Figura 2.2) è mostrato lo stato corrente del magazzino AS/RS, possono poi essere modificati in maniera software gli oggetti presenti nel buffer. A sinistra della finestra sono elencate le stazioni di lavoro contenenti un buffer; selezionandone una verranno mostrate le posizioni degli elementi nel buffer di tale stazione.

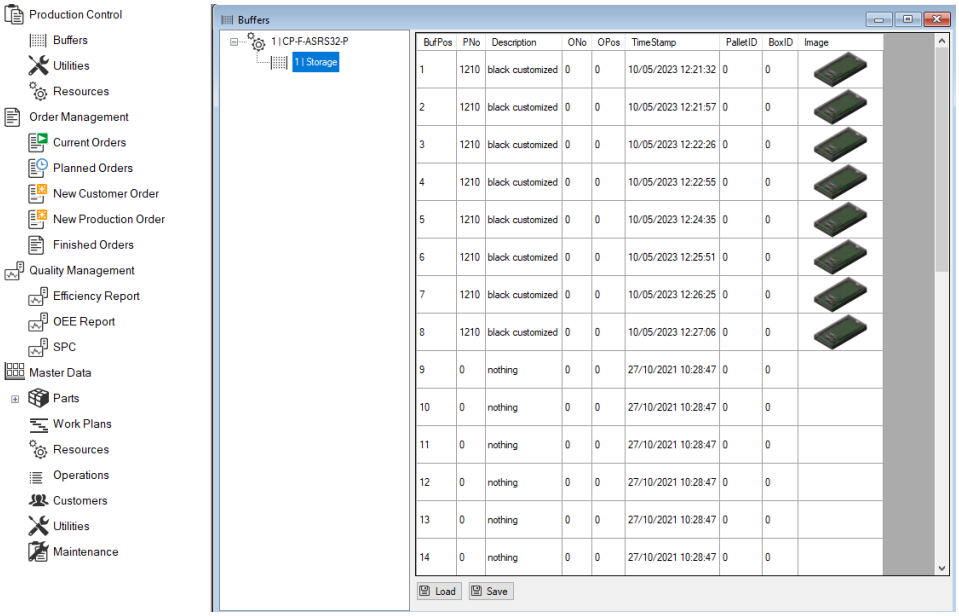


Figura 2.2.: Finestra Buffers.

Nella tabella vengono mostrate diverse informazioni sui pezzi (o elementi) all'interno del buffer:

- BufPos: numero di posizione nel buffer.

- PNo: numero identificativo del pezzo.
- Description: breve descrizione del pezzo.
- ONo: numero dell'ordine (se è 0 significa che non c'è alcun ordine per quella parte).
- OPos: posizione dell'ordine.
- TimeStamp: time stamp dell'ultima modifica di quella risorsa.
- PalletID: pallet ID del pezzo.
- BoxID: box ID del pezzo.
- Image: immagine del pezzo presente.

Nella situazione in cui, al termine della lavorazione, alcuni prodotti risultino essere da scartare, verranno raffigurati con un'immagine di prodotto non riconosciuto ed il numero identificativo PNo: 26.

Cliccando con il tasto destro del mouse su un elemento della tabella del buffer, può essere modificato:

- Edit: con questa opzione è possibile aprire una nuova finestra, dove si può cambiare il pezzo presente in quella posizione nel buffer.
- Clear: il pezzo può essere rimosso dalla sua posizione nel buffer.

Al termine di un ordine è necessario andare sulla finestra di buffer, selezionare i prodotti completati nelle loro rispettive posizioni e modificare via software i pezzi presenti nel buffer, reimpostando come pezzo sul pallet il solo guscio posteriore col numero identificativo PNo: 210. Senza questa operazione non sarà possibile effettuare nuovi ordini, dal momento che il software fino a quell'istante sa che nel buffer sono presenti solo pezzi già lavorati.

2.2.2. Production Control - Resources

Aprendo la finestra delle risorse (Figura 2.3) è possibile controllare lo stato di ogni stazione. La tabella contiene diverse informazioni:

- Picture: piccola immagine della stazione sotto controllo.
- ID: numero identificativo della risorsa (ovvero della stazione).
- Name: nome della stazione.
- MESMode: l'indicatore è verde se la modalità MES è attiva, altrimenti è rosso.
- AutomaticMode: l'indicatore è verde se la stazione è in modalità automatica, altrimenti è grigio.

- ManualMode: l'indicatore è verde se la stazione è in modalità manuale, altrimenti è grigio.
- Busy: l'indicatore è giallo se la stazione è occupata, altrimenti è grigio.
- Reset: l'indicatore è giallo se la stazione è in stato di reset, altrimenti è grigio.
- ErrorL0: l'indicatore è rosso se è presente un errore di livello 0 nella stazione, altrimenti è grigio.
- ErrorL1: l'indicatore è rosso se è presente un errore di livello 1 nella stazione, altrimenti è grigio.
- ErrorL2: l'indicatore è rosso se è presente un errore di livello 2 nella stazione, altrimenti è grigio.
- IP: indirizzo IP della stazione.
- Connected: l'indicatore è verde se la stazione è connessa al server del MES, altrimenti è rosso.

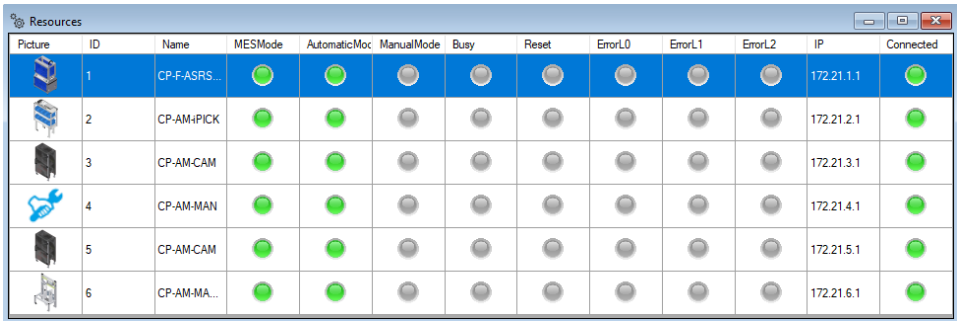


Figura 2.3.: Finestra delle Risorse.

Gli errori rilevati vengono riportati con un ordine di priorità, ovvero più il numero è basso e più l'errore è importante:

- Errori di livello 0: il programma viene immediatamente fermato e la modalità automatica viene terminata, la causa dell'errore necessita di essere risolta il prima possibile, è necessario un reset della stazione al termine della problematica.
- Errori di livello 1: il programma e la modalità automatica vengono terminati alla fine del ciclo, la causa dell'errore necessita di essere risolta, è necessario un reset della stazione al termine della problematica.
- Errori di livello 2: il programma e la modalità automatica non vengono interrotti, se la causa dell'errore viene sistemata allora l'errore viene automaticamente riconosciuto ed eliminato.

Inoltre tutti gli errori vengono poi riportati sul display HMI di ogni stazione, ma non vengono processati dal MES.

2.2.3. Order Management - Current Order

In questa finestra (Figura 2.4) è possibile controllare lo stato di ogni ordine al momento in corso. Gli ordini sono ordinati per numero d'ordine in base all'orario di inizio pianificato e ciascun ordine ha una o più posizioni d'ordine al suo interno, infatti un ordine può essere composto da più oggetti da produrre.

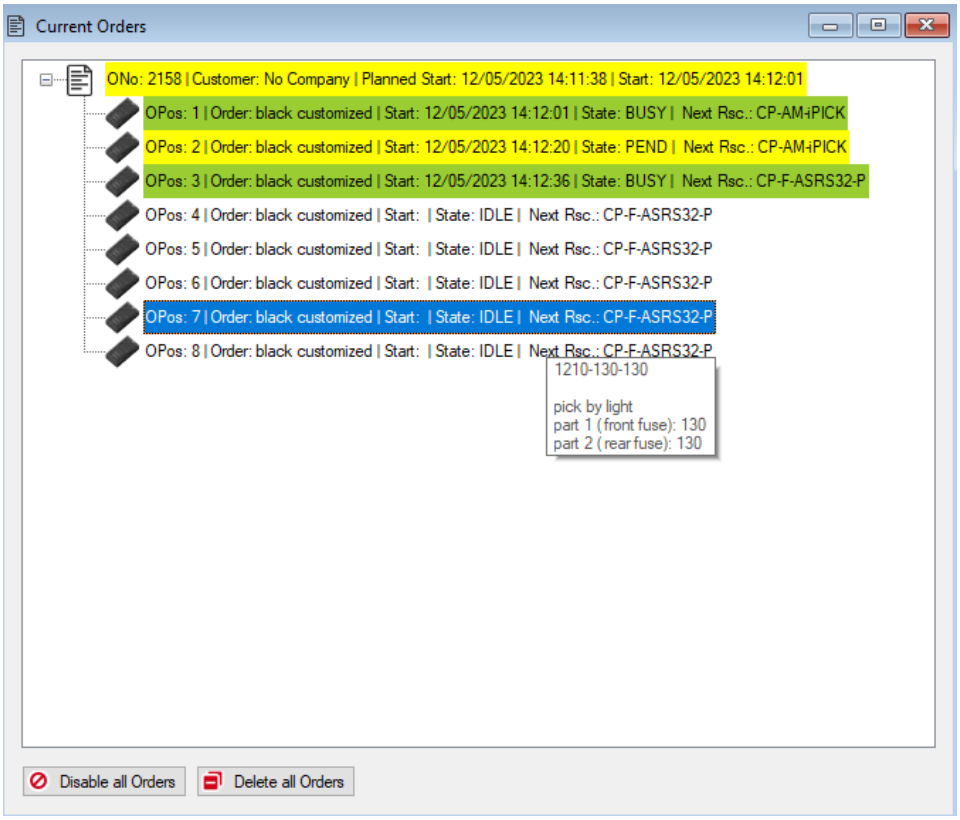


Figura 2.4.: Ordine in corso.

Una riga nella lista degli ordini contiene le principali informazioni riguardo l'ordine in corso:

- ONo: numero dell'ordine.
- Customer: il cliente dell'ordine.
- Planned Start: orario in cui la produzione dell'ordine teoricamente deve iniziare.
- Start: orario in cui la produzione dell'ordine effettivamente inizia.

Le informazioni su ogni prodotto contenuto nell'ordine sono poi mostrate nella sotto lista dell'ordine:

- OPos: numero di posizione del prodotto nell'ordine eseguito.
- Order: descrizione dell'ordine.

- Start: tempo di inizio della produzione dell'oggetto.
- State: stato dell'oggetto nel corso della produzione (PEND, IDLE, BUSY, ERROR).
- Next Rsc: la prossima stazione in cui il prodotto deve andare.

Una interfaccia che implementa dei colori diversi con significati diversi per ogni oggetto in produzione facilita la comprensione dello stato dell'ordine:

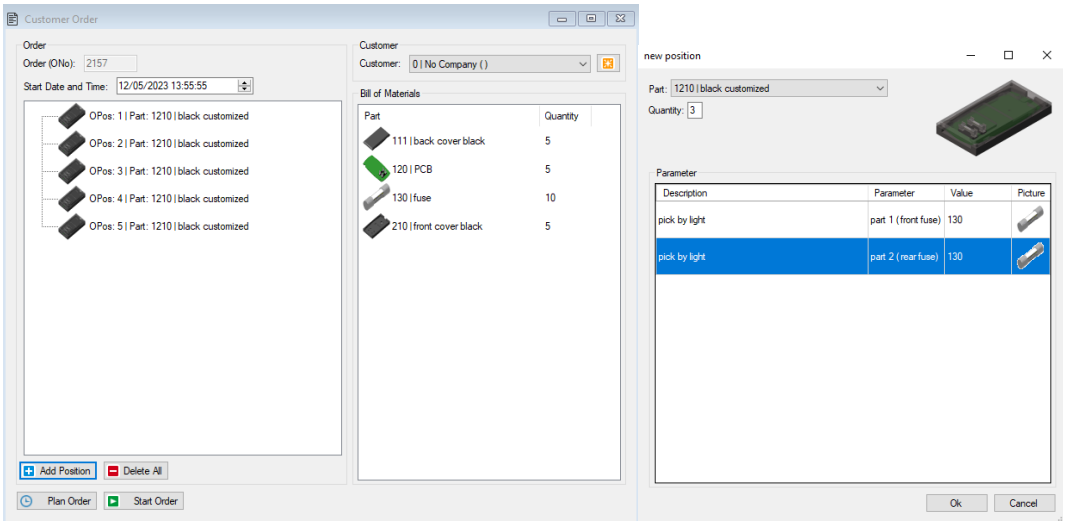
- Bianco: è lo stato normale, indica che il processamento dell'ordine non è ancora iniziato.
- Blu: l'ordine è stato selezionato dal click del mouse, oppure l'ordine è terminato.
- Giallo: la produzione dell'ordine è iniziata ed è in esecuzione.
- Lime: l'ordine è in stato di BUSY.

Selezionando un ordine in corso nella finestra degli ordini correnti è possibile:

- Disabilitare l'ordine: questa opzione è disponibile solo quando nessun prodotto contenuto nell'ordine selezionato è andato in esecuzione. Significa che l'ordine può essere spostato nuovamente nella finestra di pianificazione degli ordini, sarà possibile rieseguire l'ordine in un secondo momento.
- Eliminare l'ordine: se l'ordine viene cancellato, verranno cancellati tutti gli oggetti da produrre di quel determinato ordine.

2.2.4. Order Management - New Customer Order

In questa finestra (Figura 2.5a) è possibile effettuare la creazione di un nuovo ordine per un dato cliente. In alto a sinistra viene mostrato il numero identificativo dell'ordine, che altro non è che il numero di ordini effettuati in totale dall'applicazione. Subito sotto ad esso è presente la data di inizio dell'ordine, la quale è modificabile. Un riquadro è dedicato alla rappresentazione dei pezzi in programma per l'ordine che si sta creando. A destra troviamo poi il cliente che effettua l'ordine ed una lista dei materiali da utilizzare per la produzione dell'ordine.



(a) Finestra Customer Order. (b) Finestra New Position.

Figura 2.5.: Creazione di un nuovo ordine.

Nella parte inferiore della tabella poi sono presenti 4 pulsanti che rispettivamente indicano:

- Add Position: apertura della finestra New Position (Figura 2.5b) per l'aggiunta della tipologia di oggetto e la quantità da produrre.
- Delete All: annullamento di qualsiasi modifica effettuata nella finestra.
- Plan Order: aggiunta dell'ordine appena creato alla finestra degli ordini pianificati, dunque con un avvio della produzione pianificato.
- Start Order: esecuzione immediata dell'ordine appena creato.

2.2.5. Master Data - WorkPlan

Nella finestra di WorkPlan (Figura 2.6) è possibile creare, osservare o modificare il piano di lavoro di un prodotto che sarà poi selezionabile nella creazione degli ordini.

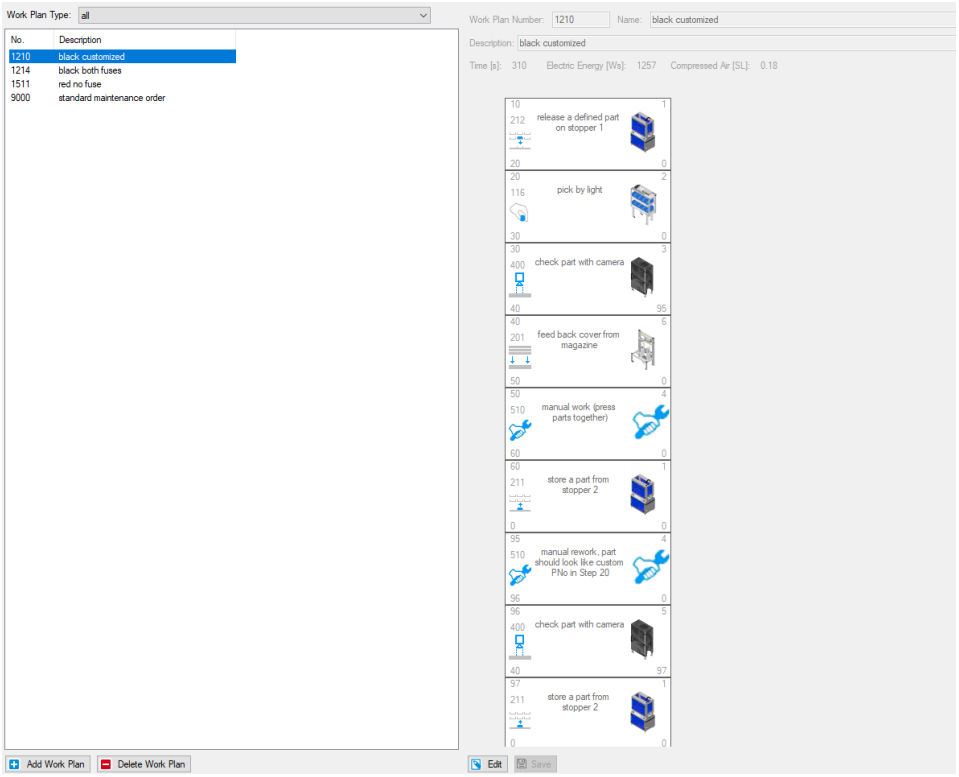


Figura 2.6.: Finestra di WorkPlan.

Viene raffigurata la sequenza di produzione del prodotto selezionato. Tale sequenza è una unione di rettangoli in cascata: l'inizio della sequenza è in alto e la fine è in basso. Ogni rettangolo (Figura 2.7) contiene le informazioni relative all'operazione che deve svolgere ed altre informazioni come:

- In alto a sinistra (30): il numero dello step.
- Poco sotto (400): il numero dell'operazione e la sua corrispondente figura.
- In basso a sinistra (40): il numero del prossimo step.
- Al centro: la descrizione dell'operazione.
- In alto a destra (3): il numero della stazione su cui viene effettuata l'operazione.
- In basso a destra (95): il numero dello step nel caso l'operazione rileva un errore.

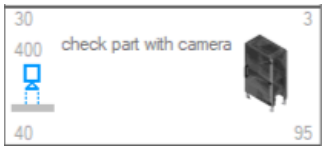


Figura 2.7.: Operazione.

Si può poi modificare un work plan dal tasto di edit della Figura 2.6. La modifica può essere effettuata sulla sequenza delle operazioni che la macchina CPFactory deve eseguire e quindi che tipologia di prodotto creare; oppure può essere effettuata sul singolo step (Figura 2.8) variando diversi parametri al fine di crearne uno nuovo o modificarlo:

- Step: numero dello step.
- First Step: viene evidenziato nel caso dovesse essere il primo step della produzione.
- Next Step: numero del prossimo step ad attivarsi dopo aver terminato in quello attuale.
- Error Step No.: numero del prossimo step nel caso di errore.
- Error Step: viene evidenziato nel caso sia uno step di errore.
- Operation: selezione dell'operazione da effettuare.
- Resource: la stazione che effettua l'operazione.
- Description: descrizione dell'operazione.
- Transport Time [s]: tempo richiesto per trasportare il prodotto fino alla stazione.
- Working Time [s]: tempo di lavorazione richiesto.
- Electric Energy : consumo di energia elettrica per questo step.
- Compressed Air: consumo di aria compressa per questo step.

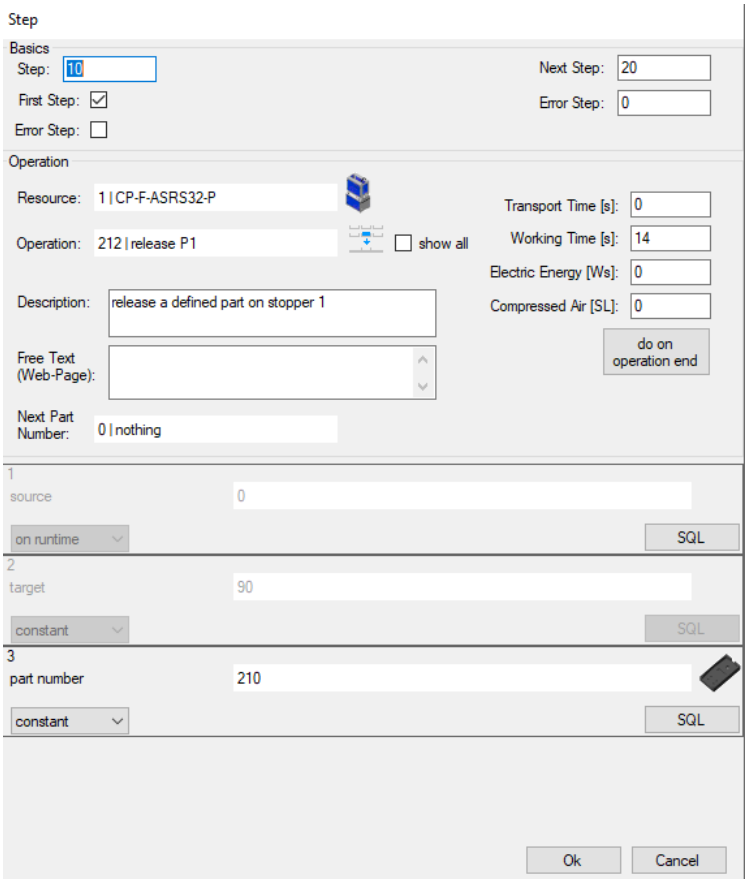


Figura 2.8.: Finestra Step.

2.3. Considerazioni sul tempo ciclo di un prodotto

Il "tempo ciclo" è il tempo che impiega un prodotto ad essere creato nel ciclo di produzione. In Figura 2.6 è presente una voce Time[s], la quale indica la somma totale di tutti i tempi inseriti negli step della sequenza per la realizzazione del prodotto. Però questa somma di tempi non è il tempo ciclo ideale, ovvero la differenza tra PlanedEnd (data di fine lavorazione pianificata) e PlanedStart (data di inizio lavorazione pianificata), di un prodotto singolo. Pertanto per il calcolo del tempo ciclo teorico ideale viene effettuata solo la somma dei tempi degli step che fanno parte della sequenza senza errori durante la produzione.

Infatti nella Figura 2.6 è possibile notare come il Time[s] della sequenza del prodotto 1210 è di 310 secondi, quando in realtà il tempo ciclo teorico ideale: per un prodotto corretto è di 218 secondi, ovvero 3 minuti e 38 secondi; per un prodotto scartato è di 140 secondi, ovvero 2 minuti e 20 secondi; per un prodotto che ha subito una lavorazione manuale errata ed il conseguente rework si ha un tempo ciclo di 266 secondi, ovvero 4 minuti e 26 secondi.

2.4. Database

Il database è aperto, basato su Microsoft Access Database, e può interfacciarsi sia in scrittura che in lettura con programmi esterni dell'utente tramite comandi SQL. All'interno del database *FestoMES* vengono raccolti tutti i dati registrati dalla macchina e utilizzati dal MES per il corretto funzionamento: sono presenti 63 tabelle di cui in seguito verrà descritto il funzionamento (Tabella 2.1).

Il database *FestoMES* inoltre non viene aggiornato periodicamente ad un intervallo di tempo regolare, ma tramite eventi, ovvero se la CPFactory non sta producendo nulla allora la macchina non registra nulla e quindi il database non viene aggiornato. Stessa cosa accade se non avviene alcun cambiamento nel programma MES, come per esempio aggiornare quale prodotto è presente nel buffer AS/RS.

Questa caratteristica è stata di fondamentale importanza per la realizzazione del calcolo dell'indice OEE, in quanto la tabella *tblMachineReport* si aggiorna solo quando c'è un nuovo evento, dunque se la macchina è senza ordini da produrre, oppure è esente da errori, non si aggiorna. Ecco perché è sufficiente prendere come tempo di produzione Tp solo quando la CPFactory è impostata in modalità *automatica*, dato che se lavora aggiorna la tabella e quando termina la lavorazione smette di aggiornarla lasciando appunto il tempo effettivo di produzione. Per ulteriori chiarimenti sui parametri relativi alla disponibilità della macchina e le diverse modalità di funzionamento è possibile visionare la Tabella 3.1 riportata nel capitolo successivo.

Le tre righe evidenziate in giallo nella tabella sottostante corrispondono alle tabelle del database utilizzate per il calcolo degli indicatori chiave di prestazione (*KPI)* che verranno descritti nel prossimo capitolo. Verranno anche spiegati più nel dettaglio i parametri di tali tabelle corrispondenti alle variabili nelle formule per il calcolo dei *KPI*.

Tabella 2.1.: Tabelle del database FestoMES

Nome	**Descrizione**
tblAvgDocked	—
tblAltOperation	—
tblAltOperationParameter	—
tblAutomaticOrder	—
tblAutoOrderType	—
tblBookedBufMsg	—
tblBox	—
tblBoxDef	—
tblBoxPos	—

Continua nella pagina successiva

Tabella 2.1.: Continuazione della tabella

Nome	**Descrizione**
tblBuffer	descrive la composizione del buffer AS/RS, ovvero da quante righe e colonne è composto il magazzino, quale è la sua funzione e l'ID della stazione
tblBufferPos	descrive, per ogni posizione nel buffer, l'oggetto che è presente sul pallet, per esempio: 210 = cover posteriore su pallet, 1210 = prodotto finale con due fusibili, 26 = pezzo sconosciuto (scarto)
tblBufferType	elenco delle possibili tipologie di buffer che ci possono essere, nel nostro caso tutto ciò che riguarda il buffer è identificato come "storage"
tblCarrier	descrizione di tutti i carrier presenti sulla macchina (nella realtà ce ne sono solo 5)
tblCarrierDef	—
tblChartDef	—
tblCustomer	descrizione dei clienti registrati
tblDataType	descrizione delle variabili utilizzate
tblErrorCodes	elenco dei codici di errore utilizzati per gli ordini
tblFinOrder	descrizione di tutti gli ordini eseguiti
tblFinOrderPos	descrizione di tutti i pezzi prodotti di tutti gli ordini eseguiti
tblFinStep	descrizione di tutti gli step che ha effettuato ciascun pezzo prodotto per tutti gli ordini eseguiti
tblFinStepParameter	descrizione dei parametri utilizzati ad ogni step
tblMachineReport	elenco del report della macchina ad ogni cambio di stato registrato in ciascuna stazione
tblMaintError	elenco dei possibili errori rilevati dalla CP_F
tblMaintErrorLinks	elenco dei link che portano a maggiori informazioni sulla natura e descrizione dell'errore

Continua nella pagina successiva

Tabella 2.1.: Continuazione della tabella

Nome	**Descrizione**
tblMaintLinks	stessa cosa della riga precedente
tblMaintSolutions	elenco delle soluzioni riportate dalla CP_F in seguito alla rilevazione dell'errore
tblMClassType	—
tblMrpType	—
tblOperation	descrizione di tutte le operazioni possibili
tblOperationParameter	—
tblOrder	—
tblOrderPos	—
tblPallet	—
tblPalletDef	—
tblPalletPos	—
tblParameterTypes	descrizione della tipologia di parametri
tblParameterValueTypes	descrizione della tipologia di valori dei parametri
tblParts	descrizione di tutti i possibili pezzi singoli e prodotti assemblati disponibili nella CP_F
tblPartsReport	elenco del report di ciascun prodotto nel corso della produzione per ogni stazione utilizzata
tblPartsType	—
tblPlcType	descrizione dei PLC utilizzati
tblPNoGroup	descrizione del numero identificativo di qualsiasi pezzo e prodotto utilizzato dalla macchina
tblNoGroupDef	raggruppamento dei pezzi e prodotti descritti nella tabella precedente in determinate categorie
tblResource	descrizione delle stazioni di lavoro presenti nella CP_F
tblResourceOperation	descrizione delle operazioni associate alle stazioni di lavoro

Continua nella pagina successiva

Tabella 2.1.: Continuazione della tabella

Nome	**Descrizione**
tblResourceType	descrizione della tipologia della stazione di lavoro (es: lavoro manuale, spostamento automatico, ecc.)
tblServiceTimetable	—
tblSqlQuery	descrizione delle query utilizzate dal software
tblSqlQueryParameter	descrizione dei parametri usati nelle query
tblSqlQueryParameterList	raggruppamento dei parametri usati nelle query nella riga precedente in determinate categorie
tblStates	descrizione del possibile stato che può assumere l'ordine in fase di esecuzione
tblStep	descrizione dello step in corso
tblStepDef	descrizione dei possibili step registrati nel MES per i prodotti (1210, 1214, 1511)
tblStepParameter	—
tblStepParameterDef	—
tblTextList	elenco dei messaggi di testo
tblTopology	—
tblTopologyType	—
tblUserControl	descrizione dell'utente che sta usando la CP_F
tblWorkPlanDef	descrizione dei work plan creati dentro il programma MES dall'utente
tblWorkPlanType	descrizione della tipologia di work plan

.

Capitolo 3.

Indicatori chiave di prestazione della produzione (KPI di produzione)

3.1. Premessa

Gli indicatori chiave di prestazione, ossia *Key Performance Indicator (KPI)*, sono indici che monitorano l'andamento e la prestazione di flussi e processi aziendali. Sono fondamentali per permettere alla produzione di avere delle "linee giuda", che da un lato garantiscono una certa conformità rispetto alle procedure da mettere in atto, dall'altro forniscono dati utili al miglioramento delle performance produttive [2].
I KPI di produzione in questo modo tengono sotto controllo l'efficienza del processo produttivo, andando ad individuare le pratiche scorrette che diminuiscono il livello di produttività e i flussi che sprecano tempo e non producono valore.
Lo scopo è quello di ottimizzare la produzione, in modo da avere meno scarti possibili, in tempi brevi e con macchinari che funzionino per tutto l'intervallo per il quale è stato programmato il loro utilizzo. In questo modo si punta a posizionarsi in modo ancora più competitivo sul mercato (Figura 3.1).

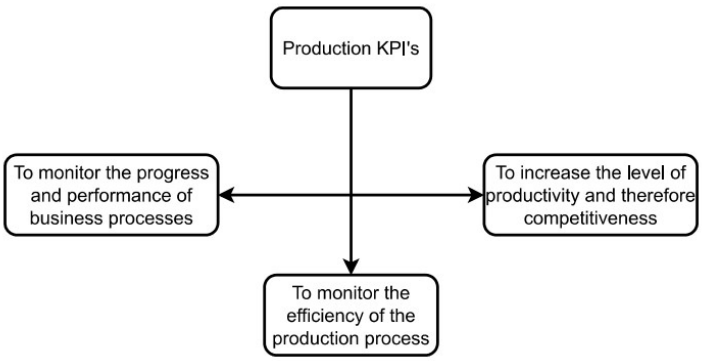


Figura 3.1.: Scopi dei KPI.

Nell'ambito di un'azienda produttiva è fondamentale individuare i giusti indicatori KPI di *efficienza*, tenendo conto di tutti gli elementi che popolano l'ecosistema aziendale. Si ritrova infatti, nel totale del lavoro prodotto e da monitorare, una componente da attribuire ai macchinari ed una da attribuire all'uomo. La proporzione

tra le due varia a seconda del tipo di azienda che si va ad analizzare, tuttavia sono quasi sempre riscontrabili entrambe.
Gli indicatori globali di efficienza più utilizzati sono dunque: per le risorse macchina l'indice OEE, per le risorse uomo l'indice OLE (ROLE) ed una valutazione dell'efficacia complessiva della produzione chiamata OTE.

Gli indici di prestazione dunque vengono calcolati in svariate situazioni da quando i produttori hanno dedicato attenzione e risorse ai miglioramenti della catena di approvvigionamento al fine di aumentare la competitività e la redditività. Per esempio Peter Dobra e János Jósvai presentano un metodo per aumentare l'OEE nelle linee di assemblaggio manuali mediante data mining [3].

Nella trattazione di questo elaborato gli indici di prestazione sono utilizzati per il monitoraggio delle prestazioni di macchinari e degli operatori umani su una linea di produzione automatizzata in scala mediante la programmazione di sistemi di automazione e software MES.

3.2. Indice OEE

L'indice OEE, acronimo che sta per *Overall Equipment Effectiveness*, è uno degli indicatori di produttività più utilizzati ed efficaci nel campo delle aziende produttive, che permette di individuare gli sprechi in produzione divisi per categoria. L'indice OEE misura in percentuale l'efficienza di una apparecchiatura di un impianto produttivo, e lo fa moltiplicando tra loro tre fattori principali: disponibilità, efficienza e qualità [4].

$$OEE = A \cdot P \cdot Q \tag{3.1}$$

- **Disponibilità (A)**: rappresenta la percentuale di *tempo programmato* effettivamente sfruttato una volta eliminati tutti i tempi di inattività dovuti a *macro-interruzioni* dell'attrezzatura, rimuovendo così le perdite causate da guasti, tempo di inattività e perdite dovute alla preparazione dell'attrezzatura.

- **Efficienza (P)**: rappresenta la percentuale di *tempo operativo* effettivamente sfruttato una volta eliminati tutti i tempi di inattività dovuti a *micro-interruzioni* dell'attrezzatura o riduzioni di velocità, inclusi l'attesa e le brevi interruzioni, la riduzione della velocità e la ridotta resa.

- **Qualità (Q)**: rappresenta la percentuale di tempo operativo netto effettivamente sfruttato una volta eliminati tutti i tempi di inattività dovuti alle attività di lavoro per il processo di unità non vendibili (ad esempio scarti di produzione o lavorazione di unità difettose).

Riprendendo l'equazione 3.1 è possibile calcolare l'OEE più nel dettaglio [5] espandendo i tre fattori principali:

$$OEE = \frac{T_u}{T_t} \cdot \left(\frac{T_p}{T_u} \cdot \frac{R_{\text{avg}}^{(a)}}{R_{\text{avg}}^{(th)}} \right) \cdot \frac{P_g}{P_a} \tag{3.2}$$

dove:

- T_u: tempo di funzionamento dell'attrezzatura (uptime).
- T_p: tempo di produzione dell'attrezzatura.
- T_t: tempo totale di osservazione dell'attrezzatura.
- $R_{\text{avg}}^{(a)}$: velocità di elaborazione effettiva media per l'attrezzatura in produzione per l'output effettivo del prodotto.
- $R_{\text{avg}}^{(th)}$: velocità di elaborazione teorica media per l'output effettivo del prodotto.
- P_g: output di prodotti di buona qualità (unità) dall'attrezzatura durante Tt.
- P_a: unità di prodotto effettivo elaborate dall'attrezzatura durante Tt.

3.2.1. Analisi delle perdite

Le perdite sono attività che assorbono risorse senza creare valore e possono essere suddivise in base alla frequenza con il quale si verificano, dalla loro causa e dalle diverse tipologie esistenti [6].
L'indice OEE è stato introdotto per la prima volta da Nakajima nel 1988 e da allora sono state riscontrate sei perdite principali da affrontare nel sistema di produzione al fine di migliorare il valore OEE [3]:

- **Guasti delle apparecchiature**: i guasti e le rotture dell'attrezzatura causano una significativa perdita di tempo e produttività all'interno di un'azienda. Questo è in genere classificato come perdita di *disponibilità* in quanto si tratta di tempi di inattività non pianificati. Quando l'attrezzatura non funziona come previsto, si verificano sprechi di tempo (che comportano un ritardo generale) e di produttività (la fabbrica produce una quantità inferiore di beni a causa dei guasti o delle rotture dell'attrezzatura).
  Questa perdita può anche includere aspetti rilevanti come guasti agli utensili, manutenzione non pianificata, mancanza di materiale o dell'operatore (lavoratore) oppure un blocco in una delle fasi del processo di produzione.
- **Configurazione e regolazione della macchina**: la seconda perdita di OEE nella produzione riguarda l'indisponibilità dell'attrezzatura a causa delle fasi di preparazione, installazione, personalizzazione o altre necessità di regolazione. Durante questo periodo, la macchina dovrebbe essere in funzione, ma invece

viene regolata (per vari motivi).
Anche se viene considerata una perdita di *disponibilità*, rientra nella categoria delle interruzioni programmate (poiché i lavoratori o i tecnici interrompono il lavoro perché devono regolare l'attrezzatura prima che riprenda a funzionare). Pulizia, manutenzione periodica, tempo di riscaldamento, tempo di raffreddamento, ispezioni di qualità, ecc., sono alcuni esempi di questo tipo di interruzioni.

- **Fermi o interruzioni minori**: le brevi interruzioni o pause sono piccoli periodi in cui l'operatore della macchina interrompe temporaneamente l'utilizzo per risolvere un problema. Di solito durano uno o due minuti (per ogni interruzione). Queste interruzioni sono categorizzate come perdita di *efficienza* in quanto l'attrezzatura è disponibile e in funzione, ma non produce durante quel breve periodo.
  Le brevi interruzioni possono includere l'attesa della macchina a causa di un leggero ritardo nell'alimentazione del materiale in ingresso, impostazioni errate, sensori non allineati, inceppamenti di materiale, complessità del design e sessioni periodiche obbligatorie di pulizia. Questi tipi di interruzioni possono essere difficili da individuare e possono portare a guasti più gravi se vengono trascurate.

- **Velocità più lente o ridotte**: questa è la durata in cui l'attrezzatura funziona più lentamente rispetto al Tempo Ciclo Ideale, con conseguente produzione di una quantità inferiore di beni rispetto a quanto previsto. È categorizzata come perdita di *efficienza* in quanto l'attrezzatura non produce tanti beni come di solito fa. Il Tempo Ciclo Ideale è il tempo più veloce possibile impiegato per produrre un prodotto. È un valore teorico calcolato in base alle specifiche fornite della macchina.
  Ci possono essere molte ragioni che portano alla riduzione della velocità. Alcune cause comuni sono una manutenzione impropria dell'attrezzatura, una scarsa lubrificazione, inceppamenti, pezzi di ricambio consumati, un ambiente di lavoro non idoneo alla produzione, materie prime o materiali di input di scarsa qualità, operatore inesperto, spegnimenti improvvisi, ecc.

- **Difetti di processo**: ogni lotto prodotto può contenere alcuni pezzi o beni difettosi nonostante il processo di produzione sia stabile. Sia i prodotti scartati che quelli che possono essere riparati o riutilizzati vengono inclusi in questa perdita. L'OEE misura la qualità in base al First-Pass Yield [7], il che rende i difetti di processo un problema di perdita di *qualità*. Se la qualità dei prodotti fosse buona, non verrebbero considerati difettosi.
  I difetti di processo possono essere causati da varie ragioni come impostazioni errate, inefficienza dell'operatore o scadenza dei materiali.

- **Minore resa o maggiore scarto**: l'ultima perdita di OEE è rappresentata dalla resa inferiore causata dai difetti di processo. La resa ridotta rientra nella categoria delle perdite di *qualità*. Sebbene possa verificarsi in qualsiasi fase della produzione, la resa ridotta o inferiore viene notata principalmente (con alta frequenza) dopo i cambi di produzione.
  La minore resa è causata da diverse ragioni, come impostazioni errate, cambi di produzione sub ottimali, implementazione errata dei cicli di riscaldamento o utilizzo di attrezzature che generano scarti dopo l'avvio.

Nonostante l'indubbia utilità di questo indicatore, esso si concentra adeguatamente sul fattore attrezzature/macchinari, trascurando ulteriori indagini in merito ai fattori della forza lavoro, dell'energia o dei materiali. Oltre ai guasti operativi legati alle attrezzature, i problemi spesso possono essere attribuiti a una serie di questioni relative alla forza lavoro che si sono accumulate mentre gli impianti funzionavano a capacità inferiore a quella ottimale. Le cause probabili includono difficoltà nella pianificazione delle risorse giuste, quando e dove sono richieste specifiche competenze. L'assenteismo di alcuni lavoratori, l'inefficacia o la mancanza di formazione che limitano la qualità e rallentano l'incremento della produzione, così come i cambiamenti di prodotto o i nuovi lanci di prodotto. Questi fattori evidenziano come il lavoro sia un elemento altrettanto critico della produzione da ottimizzare nei sistemi di produzione attuali e futuri basati sulla domanda [5].

3.3. Indice OLE

OLE è l'acronimo di *Overall Labour Effectiveness*, ossia l'efficienza generale del lavoro umano. Può anche essere definita come [5] il rapporto tra il valore aggiunto, cioè le ore di manodopera che abbiano portato alla creazione di un'effettiva produzione, e le ore/uomo disponibili, cioè le ore di manodopera complessive disponibili e pianificate per la produzione.

$$OLE = \frac{V_T}{T_t} \tag{3.3}$$

L'OLE è strettamente legato all'OEE, e quindi ne mutua anche, in linea di massima, le modalità di calcolo e definizione. Si può ottenere anche moltiplicando tra loro gli indici di disponibilità, performance e qualità [8][2].

$$OLE = A_{(OLE)} \cdot P_{(OLE)} \cdot Q_{(OLE)} \tag{3.4}$$

dove:

- **Disponibilità ($A_{(OLE)}$)**: la percentuale di tempo che gli operatori impiegano per produrre un effettivo valore. Ci sono diverse perdite che possono intaccare questo dato come:
  - assenteismo e consuetudini di utilizzo del lavoro: rientrano in questa categoria la malattia dei dipendenti, i permessi, approvati o meno, e tutte

le ore in cui il personale non è disponibile per via di riunioni, formazione o altre attività.

- scarsa programmazione: oltre alla necessità di avere orari flessibili e un planning ben definito da seguire, questo punto riguarda l'essenziale bisogno di disporre di persone con le giuste competenze e certificazioni.
- tempo indiretto: include tutti i ritardi del materiale, i tempi morti, i cambi turno e i tempi di cambio macchina.

- **Efficienza ($P_{(OLE)}$)**: la percentuale di prodotto realizzato e consegnato rispetto alle previsioni aziendali del tempo che intercorre dalla ricezione dell'ordine fino alla sua completa evasione. Le perdite ad essa correlate sono:
  - disponibilità di processi, istruzioni, strumenti e materiali: qualora uno o più di questi elementi dovesse mancare o essere carente, questo rallenterebbe l'intera produzione, o si ripercuoterebbe sulla qualità dei pezzi prodotti.
  - formazione e competenze: i dipendenti non sono in grado di svolgere correttamente i compiti loro assegnati, impedendo il completamento delle mansioni all'interno del turno di lavoro.

- **Qualità ($Q_{(OLE)}$)**: la percentuale di prodotto conforme agli standard qualitativi adottati. Rispetto a questo fattore le perdite che possono occorrere sono le seguenti:
  - conoscenza e competenza dei dipendenti: è una variabile che influisce sulla qualità della produzione. Un operatore competente è in grado di capire il funzionamento dei processi, riconoscere il ruolo della variabilità nella produzione, così come tarare gli aggiustamenti da fare per mantenere costanti i flussi. Qualora la qualità scendesse sotto lo standard richiesto, sa che è meglio interrompere la produzione. Questo genere di *know-how* riduce la quantità di scarti e di conseguenza la quantità di lavoro sprecato.
  - uso corretto di attrezzature e strumenti: i lavoratori non sempre utilizzano lo strumento giusto, al momento giusto e nel modo giusto.

Dunque l'indice OLE misura l'effetto cumulativo e l'interdipendenza di disponibilità, prestazione e qualità per individui. Al fine di valutare le fonti di perdite con il loro impatto correlato e indicare azioni di miglioramento è stato introdotto anche un altro indicatore, chiamato ROLE, che è organizzato allo stesso modo ma utilizza una diversa struttura di perdite e una nuova metodologia sia per la raccolta dei dati che per la misurazione effettiva dell'efficacia del lavoro [5].

3.4. Indice ROLE

Con il ROLE, ovvero *Revised Overall Labour Effectiveness*, si ha una nuova struttura delle perdite e una nuova metodologia per la raccolta dati e la misurazione efficace dell'efficacia del lavoro. Invece di adottare la suddivisione classica dell'OEE, qui le perdite vengono considerate in base a quattro categorie principali diverse (Figura 3.2): perdite strutturali (SL), perdite anomale (ANL), perdite guidate dalla gestione (MDL) e perdite di macchina/posto di lavoro (MWL), che a loro volta sono composte da ulteriori sottocategorie. In questo modo sistematico, è possibile valutare le cause delle perdite e il loro impatto correlato [8].

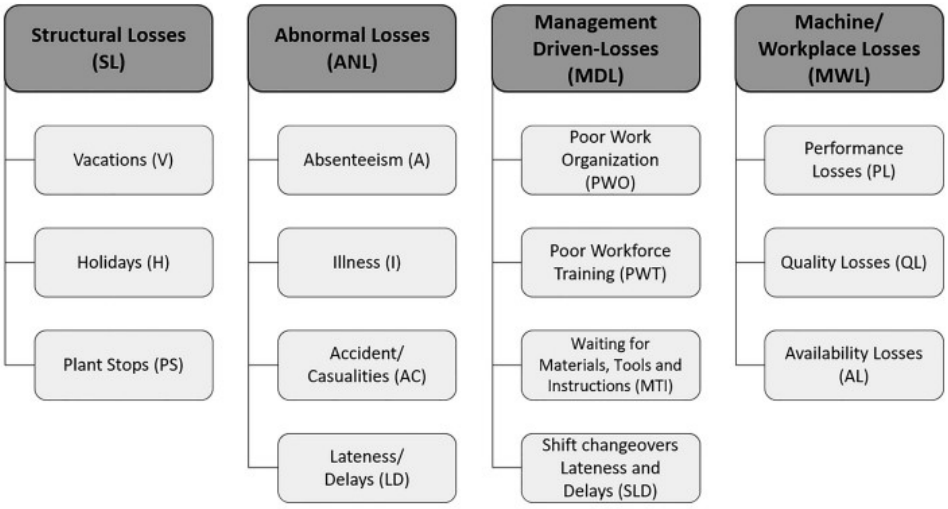


Figura 3.2.: Struttura delle perdite nel ROLE.

Questo consente, da un lato, di valutare le fonti di perdite con il loro impatto correlato e indicare azioni di miglioramento, mentre, d'altra parte, il processo di analisi delle perdite diventa piuttosto ingombrante e difficile da automatizzare. Per questi motivi, attualmente il ROLE viene proposto come strumento per aiutare il responsabile a capire se le azioni correttive hanno risolto i problemi e migliorato la produttività complessiva [5].
La Figura 3.3 presenta le relazioni temporali tra le cause di perdite nella valutazione dell'efficacia complessiva.

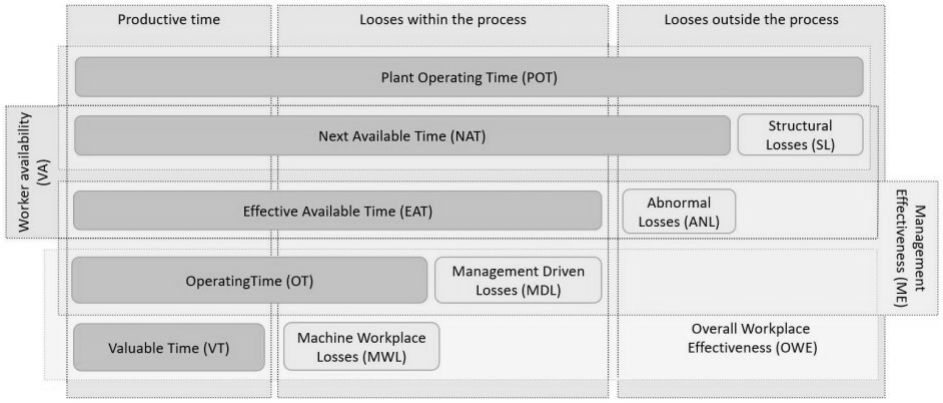


Figura 3.3.: Valutazione del ROLE.

Nella pratica, a causa di molti fattori di inefficienza (cioè perdite nascoste), riassunti nella Figura 3.2, si crea un divario tra il Tempo Valore V_T e il Tempo Teorico T_t, che deteriora progressivamente la porzione di tempo in cui un operatore può lavorare alla sua capacità nominale. Infatti, a causa di fermate programmate e non programmate, solo una parte del T_t può essere utilizzata dagli operatori per la produzione [5]. Riprendendo l'equazione 3.3 è possibile definire il ROLE come:

$$ROLE = \frac{V_T}{T_t} = \frac{V_T}{NAT} \tag{3.5}$$

dove il Tempo Netto Disponibile (NAT) viene utilizzato come Tt assumendo che solo le perdite *controllabili* possano essere affrontate mediante attività di miglioramento. Analizzando i periodi di tempo che compongono il NAT e che sono mostrati nella Figura 3.3, è possibile esprimere l'equazione 3.5 come una funzione dei tre principali fattori di inefficienza, ovvero *inefficienze del lavoratore* (OWE), *inefficienze nella gestione* (ME) e *disponibilità del lavoratore* (WA) [5].

$$ROLE = OWE \cdot ME \cdot WA \tag{3.6}$$

dove OWE è l'acronimo di Overall Workplace Effectiveness, ME sta per Management Effectiveness e WA per Worker Availability; e possono essere calcolati come segue:

$$OWE = \frac{OT - MWL}{OT} = \frac{VT}{OT} \tag{3.7}$$

$$ME = \frac{EAT - MDL}{EAT} = \frac{OT}{EAT} \tag{3.8}$$

$$WA = \frac{NAT - ANL}{NAT} = \frac{EAT}{NAT} \tag{3.9}$$

nella valutazione ROLE:
OT rappresenta il tempo operativo del lavoratore e MWL è il tempo dedicato alle

perdite dovute alla macchina o al posto di lavoro (vedi Figura 3.2);
EAT rappresenta il tempo disponibile effettivo (il tempo in cui il lavoratore è effettivamente disponibile sul posto di lavoro) e MDL è il tempo dedicato alle perdite causate dalla gestione (ad esempio, scarsa organizzazione del lavoro, formazione inadeguata del personale, attesa per materiali, strumenti e istruzioni, ritardi nella sostituzione dei turni);
ANL rappresenta il tempo dedicato alle perdite dovute a situazioni anomale (ad esempio, assenze ingiustificate, malattie, incidenti e infortuni, ritardi come ingressi e uscite tardive).

Il ROLE così definito consente di evidenziare le perdite che possono essere adeguatamente osservate e quantificate. La formulazione del ROLE consente di evidenziare le perdite in due modi diversi: nel primo caso, le perdite sono dovute al processo (OWE e ME) e a quelle non dipendenti da esso (WA). Nel secondo caso, WA è legato all'assenza del lavoratore sul posto di lavoro, mentre sia ME che OWE sono legati alle inefficienze dei lavoratori all'interno del posto di lavoro.

Il ROLE può essere ulteriormente descritto in termini di prodotti elaborati dalla forza lavoro anziché in termini di tempo, poiché è possibile assumere che il tempo di elaborazione sia direttamente proporzionale al numero di prodotti elaborati. In questo caso, la definizione del ROLE diventa [5]:

$$ROLE = \frac{P_g}{P_a^{th}} = \frac{\text{pezzi idonei nel tempo netto disponibile}}{\text{pezzi teoricamente da produrre nel tempo netto disponibile}} \tag{3.10}$$

Nello sviluppo di questo elaborato verrà utilizzata l'Equazione 3.6 per la determinazione del parametro ROLE nello script in Python, mentre per la verifica dei calcoli a mano verrà utilizzata l'Equazione 3.10.

3.5. Indice OTE

Per misurare le prestazioni ed eseguire diagnostica a livello di fabbrica esiste la metrica OTE, acronimo di *Overall Throughput Effectiveness*. Dal momento che, anche se rilevante, l'OEE non è sufficiente per valutare l'efficienza di un sistema di apparecchiature integrato e monitorare le prestazioni a livello di fabbrica, le OTE vengono applicate a sottosistemi attraverso i quali viene classificato il layout dell'intero sistema di produzione. Tali sottosistemi unici includono principalmente configurazioni in "serie", "parallelo", "assemblaggio" ed "espansione" [5].

3.5.1. Sottosistema collegato in serie

Un sottosistema connesso in serie consiste di n stazioni di lavoro individuali connesse come mostrato nella Figura 3.4.

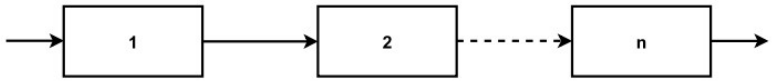


Figura 3.4.: Sottosistema connesso in serie.

Dal momento che in un sottosistema collegato in serie la stazione di lavoro con il tasso di elaborazione minimo è dominante nel processo produttivo, il tasso di elaborazione teorico di un sottosistema collegato in serie nel tempo netto disponibile NAT per l'output del prodotto effettivo (unità) è il minimo tra gli n sottosistemi, pertanto è possibile scrivere [5]:

$$OTE_{(s)} = \frac{\min\limits_{i=1,2,\ldots,n}\{\mathrm{ROLE}_{(i)} \cdot R^{\mathrm{th}}_{(i)}\}}{\min\limits_{i=1,2,\ldots,n}\{R^{\mathrm{th}}_{(i)}\}} \tag{3.11}$$

dove:

- $R^{\mathrm{th}}_{(i)}$ è il tasso di elaborazione teorico dell'$i-esimo$ sottosistema.
- $ROLE_{(i)}$ è il parametro ROLE dell'$i-esimo$ sottosistema.

3.5.2. Sottosistema collegato in parallelo

Un sottosistema connesso in parallelo consiste in n stazioni di lavoro individuali connesse come mostrato nella Figura 3.5.

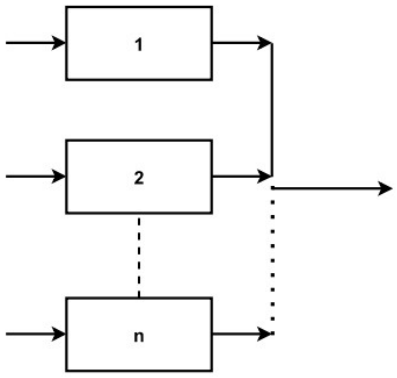


Figura 3.5.: Sottosistema connesso in parallelo.

La caratteristica che distingue l'operazione di Parallelo è associata all'indirizzamento del pezzo da uno a uno, i pezzi infatti non vengono separati su più linee ma indirizzati su una di esse per poi ricongiungersi al termine dell'operazione. Sempre durante il periodo di osservazione NAT, il calcolo utilizzato per l'indice OTE nei

sottosistemi in parallelo è invece diverso:

$$OTE_{(p)} = \frac{\sum_{i=1}^{n} \mathrm{ROLE}_{(i)} \cdot R_{(i)}^{\mathrm{th}}}{\sum_{i=1}^{n} R_{(i)}^{\mathrm{th}}} \tag{3.12}$$

3.5.3. Sottosistema collegato in assemblaggio

Un sottosistema connesso tramite una configurazione di assemblaggio consiste in n stazioni di lavoro individuali collegate tra loro come in Figura 3.6.

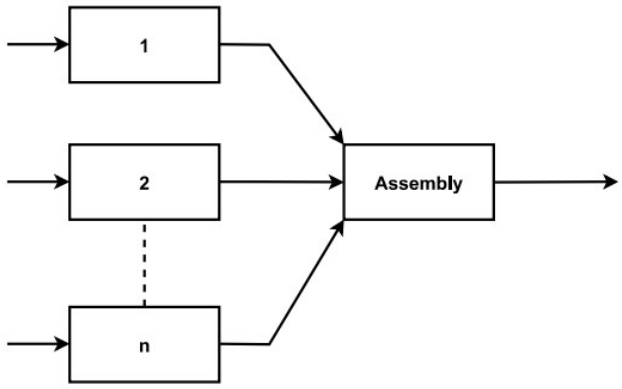


Figura 3.6.: Sottosistema connesso in assemblaggio.

La caratteristica che contraddistingue l'operazione di Assemblaggio è associata all'accorpamento dei pezzi da molti a uno, da più stazioni le varie parti che compongono un pezzo vengono unite per formare un pezzo unico. Calcolo dell'indice OTE per sottosistemi connessi in configurazione di assemblaggio [9]:

$$OTE_{(a)} = \frac{\min\left\{\min_{i=1,\ldots,n}\left\{ROLE_{(i)} \cdot \left(\frac{R_{(i)}^{th}}{K_{(i)}}\right) \cdot ROLE_{(a)}\right\}, ROLE_{(a)} \cdot R_{(a)}^{th}\right\}}{\min\left\{\min_{i=1,\ldots,n}\left\{\frac{R_{(i)}^{th}}{K_{(i)}}\right\}, R_{(a)}^{th}\right\}} \tag{3.13}$$

dove:

- $K_{(i)}$ è il numero dei pezzi richiesti dall'$i-esimo$ sottosistema per realizzare il prodotto finale della stazione di Assembly.
- $ROLE_{(a)}$ è l'indice ROLE per il sottosistema Assembly.
- $R_{(a)}^{th}$ è il tasso di elaborazione teorico del sottosistema Assembly.

3.5.4. Sottosistema collegato in espansione

Un sottosistema connesso tramite una configurazione di espansione consiste in n stazioni di lavoro individuali collegate tra loro come in Figura 3.7.

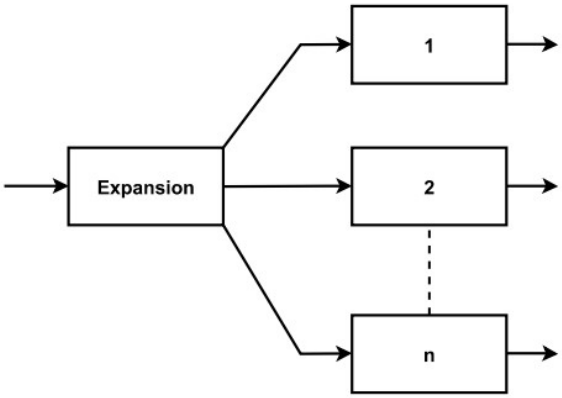


Figura 3.7.: Sottosistema connesso in espansione.

La caratteristica che contraddistingue l'operazione di Espansione è associata alla separazione dei pezzi da uno a molti, da una singola stazione il pezzo viene scomposto in più parti per essere suddiviso nelle rispettive stazioni. Calcolo dell'indice OTE per sottosistemi connessi in configurazione di espansione [9]:

$$OTE_{(e)} = \frac{\sum_{i=1}^{n} \min\left\{ROLE_{(e)} \cdot R^{th}_{(e)} \cdot K_{(i)} \cdot ROLE_{(i)}, ROLE_{(i)} \cdot R^{th}_{(i)}\right\}}{\sum_{i=1}^{n} \min\left\{R^{th}_{(e)} \cdot K_{(i)}, R^{th}_{(i)}\right\}} \tag{3.14}$$

dove:

- $K_{(i)}$ è il numero dei pezzi richiesti dall'$i-esimo$ sottosistema per realizzare il prodotto finale della stazione di Expansion.
- $ROLE_{(e)}$ è l'indice ROLE per il sottosistema Expansion.
- $R^{th}_{(e)}$ è il tasso di elaborazione teorico del sottosistema Expansion.

3.6. Mappatura dei KPI nei parametri del processo CPFactory

Per il calcolo dell'OEE e del ROLE è stato necessario mappare i parametri delle equazioni descritte sopra nei parametri del processo CPFactory resi disponibili dalle stazioni tramite MES.

3.6.1. Parametri dell'OEE reperibili su CPFactory

In particolare per l'indice OEE, seguendo la formula 3.2, nella tabella *tblMachineReport* del database della Festo sono presenti i valori da associare ai parametri della disponibilità elencati nella Tabella 3.1.

Parametro OEE	**Associazione al valore nel database**
Tempo totale di osservazione dell'attrezzatura (Tt)	differenza dei tempi di inizio e fine dell'intervallo impostato.
Tempo di funzionamento dell'attrezzatura (Tu)	differenza tra il tempo totale Tt e la somma dei tempi quando è nei casi: MANUAL/ERROR/L0/L1/L2.
Tempo di produzione dell'attrezzatura (Tp)	somma dei *TimeStamp* solo quando è attiva la modalità AUTOMATICA.

Tabella 3.1.: Mappatura parametri per la disponibilità.

Nella tabella *tblFinOrderPos* del database della Festo sono presenti i valori da associare ai parametri dell'efficienza elencati nella Tabella 3.2.

Parametro OEE	**Associazione al valore nel database**
Tempo di funzionamento pianificato ($R_{\text{avg}}^{(th)}$)	ricavo del tempo pianificato dalla differenza di *PlanedEnd* e *PlanedStart*.
Tempo di funzionamento reale ($R_{\text{avg}}^{(a)}$)	ricavo del tempo reale dalla differenza di *End* e *Start*.

Tabella 3.2.: Mappatura parametri per l'efficienza.

Inoltre, sempre nella tabella *tblFinOrderPos* del database della Festo, sono presenti i valori da associare ai parametri della qualità elencati nella Tabella 3.3.

Parametro OEE	**Associazione al valore nel database**
Pezzi buoni prodotti in Tt (Pg)	se il valore è *false* nella colonna *Error* della tabella, indica che la produzione del pezzo è andata a buon fine.
Pezzi totali prodotti in Tt (Pa)	somma di tutti i pezzi prodotti da tutti gli ordini effettuati.

Tabella 3.3.: Mappatura parametri per la qualità.

Una cosa importante da sottolineare è che la tabella *tblMachineReport* si aggiorna solo quando c'è un nuovo evento, dunque se la macchina è senza ordini da produrre, oppure è esente da errori, non si aggiorna. Ecco perché è sufficiente registrare il tempo di produzione (Tp) solo quando è in modalità *automatica*, dato che se la

CPFactory sta producendo aggiorna la tabella e quando termina la lavorazione smette di aggiornarla lasciando appunto il tempo effettivo di produzione.

3.6.2. Parametri del ROLE reperibili su CPFactory

Per applicare la teoria dell'indice ROLE alla CPFactory è prima necessario fare delle considerazioni sulle perdite, seguendo l'Equazione 3.6 definita come una funzione dei tre principali fattori di inefficienza, solo alcuni tra i parametri utilizzati sono stati ricavati dal database della Festo.

Dal momento che non tutte le perdite sono ricavabili dalle informazioni presenti nel database, è stato creato apposta un file di testo *input.txt* il quale contiene dati che vengono utilizzati dallo script al momento dell'esecuzione.
Le perdite strutturali SL hanno valore pari a 0, dato che sono state fatte le ipotesi che solo le perdite *controllabili* possono essere affrontate mediante attività di miglioramento. Di conseguenza osservando la Figura 3.3 è possibile affermare che $NAT = POT$, dunque il Tempo Netto Disponibile è pari al Tempo Operativo dell'Impianto ed equivale al turno di lavoro impostabile nel file di testo *input.txt*.
Anche le perdite anomale ANL possono essere inserite nell'apposito file di testo *input.txt* per registrarle nel programma.
Le restanti perdite descritte in Tabella 3.4 sono state ricavate indirettamente dal database della Festo usando la tabella *tblFinStep*.

Parametro ROLE	**Associazione al valore nel database**
Perdite causate dalla gestione (MDL)	ricavate dalla differenza tra la somma dei tempi di lavorazione reale (differenza tra *End* e *Start*) per quella data stazione di lavoro (*ResourceID*) e la somma dei tempi di lavorazione pianificati (differenza tra *PlanedEnd* e *PlanedStart*) sempre per quella data stazione di lavoro.
Perdite dovute alla macchina o al posto di lavoro (MWL)	ricavate dalla media del tempo di lavorazione reale (somma delle differenze tra *End* e *Start* diviso per il numero di pezzi lavorati) e moltiplicato per il numero di pezzi errati rilevati dal controllo qualità in *ErrorRetVal*.
Tempo operativo del lavoratore (OT)	equivale alla somma dei tempi di lavorazione pianificati (differenza tra *PlanedEnd* e *PlanedStart*) per quella data stazione di lavoro selezionata in *ResourceID*.
Tempo disponibile effettivo (EAT)	equivale alla somma dei tempi di lavorazione reale (differenza tra *End* e *Start*) per quella data stazione di lavoro selezionata in *ResourceID*.

Tabella 3.4.: Mappatura parametri per indice ROLE.

Capitolo 4.

Prove sperimentali

In questo capitolo vengono elencate una serie di prove effettuate sulla CPFactory che vanno a simulare casi in cui la macchina è in funzione e produce valore. Lo scopo di tali prove è verificare la correttezza dei risultati in uscita dallo script in Python, che verrà spiegato nel capitolo successivo.
Vengono quindi messi a confronto due valori di OEE ricavati in maniera diversa per ogni prova esposta, il test preso in considerazione verrà valutato con esito positivo se questi due valori saranno simili o uguali.

Per la determinazione dell'indice OEE in maniera automatica è stata utilizzata l'Equazione 3.2 descritta nel Capitolo 3, i parametri utilizzati da tale equazione sono descritti dalle Tabelle: 3.1, 3.2, 3.3 del Capitolo 3.
I risultati ottenuti in automatico dallo script verranno poi messi a confronto con dei calcoli *manuali* per l'OEE, questi ultimi sviluppati seguendo formule più tradizionali utilizzate dalle aziende.

Il ricavo dell'indice OEE per la verifica manuale avviene utilizzando l'Equazione 3.1 descritta sempre nel Capitolo 3, dove però cambiano le modalità di calcolo dei parametri di disponibilità, efficienza e qualità.
Vengono dunque definite le formule utilizzate per i calcoli *manuali*, per ricavare rispettivamente la *Disponibilità* (4.1), *Efficienza* (4.2) e *Qualità* (4.3):

$$A = \frac{\text{uptime}}{\text{tempo totale}} \tag{4.1}$$

$$E = \frac{\text{numero pezzi reali prodotti}}{\text{numero pezzi teoricamente da produrre}} \tag{4.2}$$

$$Q = \frac{\text{numero pezzi corretti prodotti}}{\text{numero pezzi totali prodotti}} \tag{4.3}$$

Inoltre, prima di continuare, è necessario definire come si ricava il *numero dei pezzi teoricamente da produrre*, che per semplicità verrà definito come $Pz_{num}^{(th)}$:

$$Pz_{num}^{(th)} = \frac{\text{turno di lavoro}}{\text{tempo ciclo di un pezzo}} \tag{4.4}$$

ed il *tempo ciclo* che equivale al tempo che impiega l'impianto a produrre un singolo

pezzo:

$$\text{tempo ciclo} = \frac{\text{tempo di produzione}}{\text{numero pezzi reali prodotti}} \tag{4.5}$$

4.1. Prova 1 - Caso ideale

Questa prova, effettuata in data 10/05/2023 alle 11:12, è stata nominata "caso ideale" perché durante la produzione non sono avvenuti ritardi di alcun genere o pezzi scartati a fine produzione. Infatti esegue un normale ciclo di produzione di 8 pezzi in un turno di lavoro di 30 minuti (T_t). Il tempo di produzione (T_p) è di 8:18 minuti, il tempo di fermo della macchina è circa 0, i pezzi totali (P_a) e quelli idonei (P_g) sono gli stessi.

$$\text{tempo ciclo} = \frac{8:18}{8} = 1:02'$$
$$Pz_{\text{num}}^{(th)} = \frac{30'}{1:02'} \approx 29 \text{ pz}$$

Calcolo manuale	**Calcolo automatico con script**
$A = \frac{30'}{30'} = 1$	$A = 0,98$
$E = \frac{8}{29} = 0,28$	$E = 0,26$
$Q = \frac{8}{8} = 1$	$Q = 1$
$OEE = 1 \cdot 0,28 \cdot 1 = 0,28$	$OEE = 0,25$

Tabella 4.1.: Prova 1 - Caso ideale.

La *disponibilità (A)* nello script in Python non è al 100% perché durante le prove un sensore è spesso andato in errore facendo registrare qualche secondo di fermo macchina. Il "caso ideale" è inteso per i valori di disponibilità e qualità che assume la macchina, ovviamente il parametro di efficienza non può essere elevato se la lavorazione è stata di 8 minuti su 30' osservati.

4.2. Prova 2 - Presenza di pezzi di scarto

Questa prova, effettuata in data 10/05/2023 alle 11:44, esegue un normale ciclo di produzione di 8 pezzi in un turno di lavoro di 30 minuti (T_t). Il tempo di produzione (T_p) è di 7:53 minuti, il tempo di fermo della macchina è circa 0, i pezzi totali (P_a) sono 8 mentre i pezzi idonei (P_g) sono 6 (ecco perché è stata soprannominata "presenza di pezzi di scarto"). Il motivo per il quale il tempo di produzione è inferiore rispetto alla prova precedente è perché i pezzi scartati hanno una permanenza sulla linea minore di un pezzo idoneo.
Il tempo ciclo corretto da considerare è quello della sezione 4.1 in quanto può essere considerato come *tempo ciclo ideale*, questa assunzione verrà fatta per tutte le prove

a seguire. Di conseguenza anche il numero dei pezzi teoricamente da produrre in quel turno di lavoro ($Pz_{\text{num}}^{(th)}$) rimane invariato.

Calcolo manuale	**Calcolo automatico con script**
$A = \frac{30'}{30'} = 1$	$A = 0,99$
$E = \frac{8}{29} = 0,28$	$E = 0,27$
$Q = \frac{6}{8} = 0,75$	$Q = 0,75$
$OEE = 1 \cdot 0,28 \cdot 0,75 = 0,21$	$OEE = 0,20$

Tabella 4.2.: Prova 2 - Presenza di pezzi di scarto.

Dai risultati appena raffigurati è possibile notare come, rispetto alla prova "caso ideale", il parametro di qualità in entrambi i metodi di calcolo è inferiore.

4.3. Prova 3 - Presenza di Fermo macchina

Questa prova, effettuata in data 12/05/2023 alle 14:11, possiede come differenza sostanziale rispetto al caso ideale una riduzione della disponibilità della CPFactory data da fermi macchina programmati. Viene eseguito un normale ciclo di produzione di 8 pezzi in un turno di lavoro di 30 minuti (T_t). Il tempo di produzione (T_p) è di 9 minuti, il tempo di fermo della macchina è di 15 minuti, i pezzi totali (P_a) e quelli idonei (P_g) coincidono. In questa prova è importante notare come il tempo di fermo incide negativamente sulla *disponibilità* della macchina.

$$Pz_{\text{num}}^{(th)} = \frac{30' - 15'}{1:02'} \approx 14 \text{ pz}$$

Calcolo manuale	**Calcolo automatico con script**
$A = \frac{15'}{30'} = 0,50$	$A = 0,52$
$E = \frac{8}{14} = 0,57$	$E = 0,58$
$Q = \frac{8}{8} = 1$	$Q = 1$
$OEE = 0,50 \cdot 0,57 \cdot 1 = 0,29$	$OEE = 0,30$

Tabella 4.3.: Prova 3 - Presenza di fermo macchina.

A seguito di questa prova è possibile affermare che una riduzione della disponibilità della macchina porta ad una conseguente diminuzione dell'indice OEE, ovvero la produzione nel complesso risulta meno efficiente.

4.4. Prova 4 - Produzione maggiorata

Questa prova, effettuata in data 17/05/2023 alle 10:07, è stata svolta al fine di dimostrare come con una produzione maggiore nel tempo di osservazione considerato

porta ad un conseguente aumento del parametro di efficienza e quindi ad un aumento complessivo dell'OEE. Esegue quindi un normale ciclo di produzione di 24 pezzi in un turno di lavoro di 30 minuti (T_t). Il tempo di produzione (T_p) è di 24:41 minuti, il tempo di fermo della macchina è pari a 0, i pezzi totali (P_a) sono 24 mentre quelli idonei (P_g) sono 21.

Calcolo manuale	**Calcolo automatico con script**
$A = \frac{30'}{30'} = 1$	$A = 1$
$E = \frac{24}{29} = 0,83$	$E = 0,84$
$Q = \frac{21}{24} = 0,88$	$Q = 0,88$
$OEE = 1 \cdot 0,83 \cdot 0,88 = 0,73$	$OEE = 0,74$

Tabella 4.4.: Prova 4 - Produzione maggiorata.

In questa prova è importante notare come nel tempo osservato (30 minuti) la macchina è rimasta in produzione quasi per tutto il tempo, dunque è corretto riscontrare un aumento del valore di *efficienza*.

4.5. Prova 5 - Caso misto

Questa prova, effettuata in data 17/05/2023 alle 11:03, rispetto al caso ideale contiene caratteristiche miste descritte nelle prove precedenti. Infatti esegue un normale ciclo di produzione di 16 pezzi in un turno di lavoro di 30 minuti (T_t). Il tempo di produzione (T_p) è di 16:32 minuti, il tempo di fermo della macchina è di 10 minuti, i pezzi totali (P_a) sono 16 mentre quelli idonei (P_g) sono 12.

$$Pz_{\text{num}}^{(th)} = \frac{30' - 10'}{1:02'} \approx 19 \text{ pz}$$

Calcolo manuale	**Calcolo automatico con script**
$A = \frac{20'}{30'} = 0,66$	$A = 0,65$
$E = \frac{16}{19} = 0,84$	$E = 0,86$
$Q = \frac{12}{16} = 0,75$	$Q = 0,75$
$OEE = 0,66 \cdot 0,84 \cdot 0,75 = 0,41$	$OEE = 0,42$

Tabella 4.5.: Prova 5 - Caso misto.

Anche in questo caso i valori ricavati adottando i due metodi di calcolo per l'OEE sono coerenti tra loro. Questo conferma ulteriormente la validità delle funzioni utilizzate nel codice Python che verrà introdotto nel capitolo successivo.

4.6. Prova 6 e 7 con ROLE

Prima di introdurre i calcoli sperimentali delle due prove, è necessario ridefinire l'equazione per l'indice ROLE utilizzata nei calcoli *manuali* da mettere poi a confronto con i risultati ricavati dallo script in Python, che invece utilizza l'Equazione 3.6 espressa nel capitolo precedente e i parametri ad esso associati descritti nella Tabella 3.4 del Capitolo 3.

Come già menzionato, il ROLE può essere ulteriormente descritto in termini di prodotti elaborati dalla forza lavoro anziché in termini di tempo, poiché è possibile assumere che il tempo di elaborazione sia direttamente proporzionale al numero di prodotti elaborati. In questo caso, la definizione del ROLE diventa l'Equazione 3.10 inserita nel Capitolo 3.

La prova 6, effettuata in data 17/05/2023 alle 11:37, esegue un ciclo di produzione di 8 pezzi ma con tempi di lavorazione nelle stazioni manuali esageratamente lunghi in un turno di lavoro di 30 minuti (T_t). Infatti il tempo di produzione (T_p) è di 19:44 minuti, il tempo di fermo della macchina è circa 0, i pezzi totali (P_a) e quelli idonei (P_g) coincidono. Le perdite anomale ANL sono assenti, dunque osservando la Figura 3.3 si può stabilire per questa prova che $NAT = EAT$, dove EAT equivale alla somma dei tempi di lavorazione reale per la stazione di lavoro osservata. In questa prova la media di lavorazione nella stazione Pick By Light è stata di 44 secondi per pezzo (idealmente dovrebbe essere 9 secondi, viene definito nel MES), dunque seguendo l'equazione 3.10 si ha:

$$NAT_{PickByLight} = 44'' \cdot 8 = 352''$$
$$P_a^{th} = \frac{NAT_{PickByLight}}{\text{tempo ciclo PBL}} = \frac{352''}{9''} \approx 39 \text{ pz}$$

ed una media di lavorazione nella stazione Rework di 4 secondi per pezzo (idealmente dovrebbe essere 4 secondi, viene definito nel MES), dunque secondo l'equazione 3.10 si ha:

$$NAT_{Rework} = 4'' \cdot 8 = 32''$$
$$P_a^{th} = \frac{NAT_{Rework}}{\text{tempo ciclo RW}} = \frac{32''}{4''} = 8 \text{ pz}$$

Calcolo manuale	**Calcolo automatico con script**
$A = \frac{30'}{30'} = 1$	$A = 0,97$
$E = \frac{8}{29} = 0,27$	$E = 0,23$
$Q = \frac{8}{8} = 1$	$Q = 1$
$OEE = 1 \cdot 0,27 \cdot 1 = 0,27$	$OEE = 0,22$
$ROLE_{PickByLight} = \frac{8}{39} = 0,20$	$ROLE_{PickByLight} = 0,19$
$ROLE_{Rework} = \frac{8}{8} = 1$	$ROLE_{Rework} = 0,96$

Tabella 4.6.: Prova 6 - Lunga lavorazione.

La prova 7, effettuata in data 24/05/2023 alle 11:50, esegue un ciclo di produzione di 8 pezzi con una media dei tempi di lavorazione nelle due stazioni manuali prossime al caso ideale. Il turno di lavoro è pari a 30 minuti (T_t), il tempo di produzione (T_p) è di 9:03 minuti, il tempo di fermo della macchina è circa 0, i pezzi totali (P_a) e quelli idonei (P_g) coincidono al termine della lavorazione, ma non coincidono nella stazione Pick By Light in quanto su 8 pezzi totali solo 7 sono idonei.
Valgono le stesse considerazioni della prova 6 in merito al tempo netto disponibile NAT. La media di lavorazione nella stazione Pick By Light è di circa 11 secondi per pezzo, mentre la media di lavorazione nella stazione Rework è di 4 secondi per pezzo.

$$
\begin{aligned}
NAT_{PickByLight} &= 11'' \cdot 8 = 88'' \\
P_a^{th} &= \frac{NAT_{PickByLight}}{\text{tempo ciclo PBL}} = \frac{88''}{9''} \approx 9 \text{ pz} \\
NAT_{Rework} &= 4'' \cdot 8 = 32'' \\
P_a^{th} &= \frac{NAT_{Rework}}{\text{tempo ciclo RW}} = \frac{32''}{4''} \approx 8 \text{ pz}
\end{aligned}
$$

Calcolo manuale	**Calcolo automatico con script**
$A = \frac{30'}{30'} = 1$	$A = 0,99$
$E = \frac{8}{29} = 0,27$	$E = 0,28$
$Q = \frac{8}{8} = 1$	$Q = 1$
$OEE = 1 \cdot 0,27 \cdot 1 = 0,27$	$OEE = 0,28$
$ROLE_{PickByLight} = \frac{7}{9} = 0,77$	$ROLE_{PickByLight} = 0,71$
$ROLE_{Rework} = \frac{8}{8} = 1$	$ROLE_{Rework} = 0,97$

Tabella 4.7.: Prova 7 - Normale lavorazione.

Dalle ultime due prove è possibile notare come un rallentamento di produzione nella stazione di lavorazione manuale incida sull'indice di efficienza finale. Inoltre viene verificato che l'Equazione 3.10 è sufficiente per determinare il ROLE per una specifica stazione di lavoro manuale. Rimane comunque più accurata l'Equazione 3.6

usata nello script in Python in quanto è caratterizzata da tutte le perdite osservate sulla macchina che vanno ad incidere sulla produzione.

Capitolo 5.

Implementazione degli indicatori chiave di prestazione

Il software sviluppato permette di ricavare gli indicatori chiave di prestazione ai fini di monitorare la produzione della CPFactory. Per la scrittura del codice è stato utilizzato il linguaggio di programmazione *Python*, di conseguenza un ambiente di sviluppo adeguato è *PyCharm*. Quest'ultimo è un ambiente di sviluppo integrato (IDE) destinato specificatamente alla programmazione nel linguaggio Python.

Noto lo scopo del software, per una corretta esecuzione del codice nel progetto è necessario includere il database dal quale vengono prelevate le informazioni relative alla CPFactory, un metodo per il salvataggio dei valori letti dal database e un file di testo da cui è possibile inserire le informazioni esterne come: il turno di lavoro da impostare e le perdite anomale (ANL) le quali, come indicato nel Capitolo 3, non è possibile ricavare dal momento che in esse sono comprese malattie o infortuni del lavoratore.

Nella Figura 5.1 viene mostrata l'effettiva suddivisione del progetto nei rispettivi file: *main.py* e *date_new.py* sono gli unici che contengono codice eseguibile, il database della Festo dal quale vengono prelevati i dati (nel computer locale dei laboratori i-Labs sarà nella cartella `C:\MES4\FestoMES.accdb`), il file delle impostazioni iniziali *input.txt* e le tre tabelle *.csv* utilizzate durante l'esecuzione del codice.

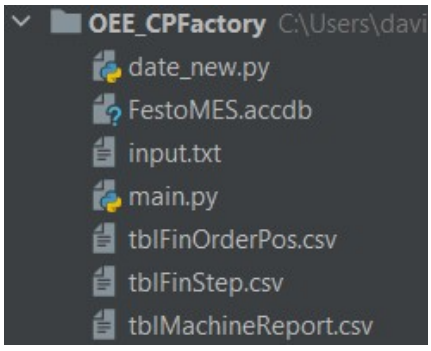


Figura 5.1.: Progetto.

5.1. Main

Nella Figura 5.2 viene mostrato tramite macro funzioni il diagramma di flusso del codice presente nel *main.py*, tale diagramma ha lo scopo di introdurre il lettore al funzionamento generale dello script. A seguito dell'avvio del programma, vengono inizializzate le configurazioni iniziali ed effettuata la chiamata al database della CPFactory. In seguito avvengono le elaborazioni dei dati appena ricavati dalle tre tabelle elencate nel diagramma. Successivamente vengono ricavati gli indicatori chiave di prestazione (KPI) relativi alla produzione avvenuta nel periodo di tempo osservato. Infine tutti i dati appena calcolati vengono salvati su un nuovo database. Verranno poi descritti maggiormente nel dettaglio i blocchi di codice per lo sviluppo del progetto.

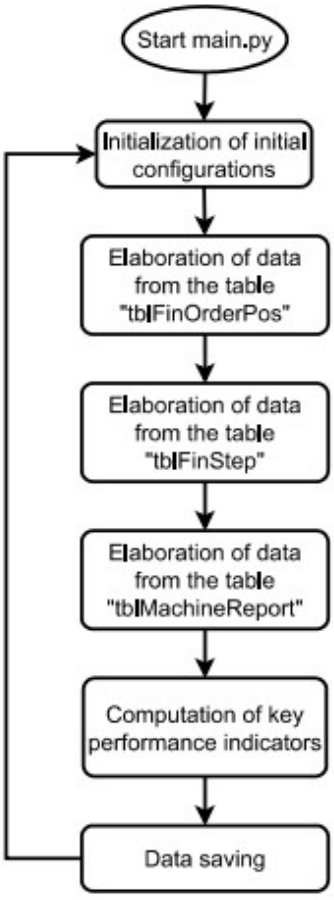


Figura 5.2.: Diagramma di flusso.

Nelle prime righe dello script sono presenti tutte le inclusioni necessarie per la corretta esecuzione del codice, ovvero tutte le librerie utilizzate.

```
import csv
import time
from datetime import datetime
from importlib import reload
import pandas as pd
import pyodbc
import mysql.connector
import date_new
```

Figura 5.3.: Inclusioni.

Le librerie *pyodbc* e *mysql.connector* sono state utilizzate per le operazioni con i due database, rispettivamente per il database *FestoMES.accdb* e per il database *MyDatabase* creato per salvare i nuovi dati. Sono state utilizzate due librerie diverse perché la prima libreria consente di effettuare operazioni su database in formato Access, mentre la seconda libreria permette l'esecuzione di chiamate a database MySQL. La scelta di sfruttare due database è dovuta dal fatto che si è preferito separare i dati grezzi letti dalla macchina in sé da quelli ricavati attraverso il processamento del software. Una diversa implementazione altrettanto valida poteva essere di creare semplicemente una nuova tabella nello stesso database della CPFactory.

Le librerie *csv* e *pandas* sono state rispettivamente usate per la scrittura dei dati letti dal database su file *.csv* e per la lavorazione dei dati attraverso l'utilizzo di dataframe. Un dataframe è una struttura dati tabulare bidimensionale, potenzialmente eterogenea, con righe e colonne etichettate.

Il modulo *date_new* e la libreria *reload* sono stati utilizzati invece per il setting del periodo temporale in cui prelevare i dati dal database ed eseguire i diversi calcoli. Inoltre, la seconda libreria contiene un metodo che permette di ricaricare i dati delle variabili presenti nel modulo *date_new* in modo tale da permettere il continuo aggiornamento del periodo temporale senza mai interrompere il codice in esecuzione.

Le restanti librerie invece sono state utilizzate per la gestione delle date e dei tempi durante lo sviluppo dello script.

Le sezioni a seguire hanno lo scopo di descrivere più nel dettaglio i blocchi delle macro funzioni raffigurate nella Figura 5.2.

5.1.1. Inizializzazione impostazioni iniziali

In questa sezione viene spiegato il blocco "inizializzazione impostazioni iniziali" annunciato nel paragrafo precedente, seguendo lo schema in Figura 5.4.

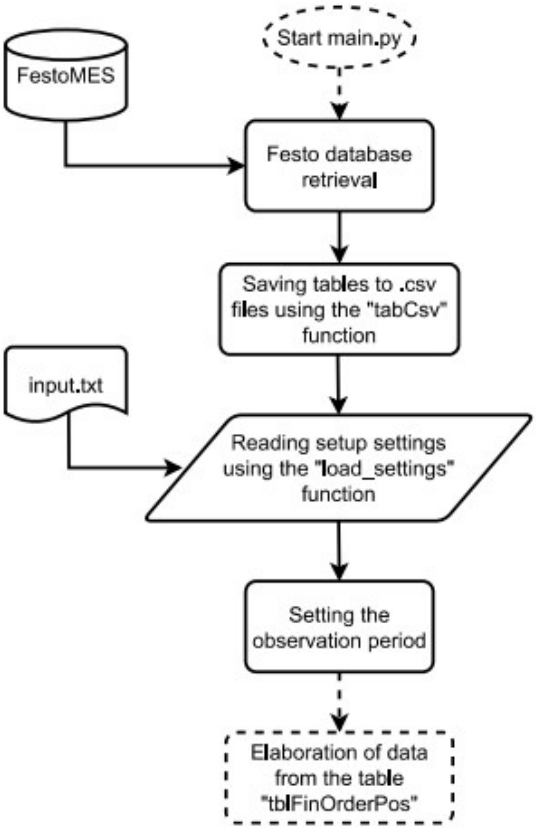


Figura 5.4.: Diagramma di flusso del blocco di inizializzazione.

Il primo step che viene eseguito è la chiamata al database della Festo tramite la funzione *"call_to_database()"*, successivamente i dati appena letti dalle tabelle del database vengono salvati su delle tabelle *.csv* attraverso la funzione *"tabCsv(titolo,results)"*. In seguito il codice legge dal file di testo *input.txt* alcune impostazioni iniziali che devono essere aggiornate prima di avviare lo script, come turno di lavoro e perdite anomale (ANL). La funzione *"load_settings(path_file)"* si occupa di leggere tale file e salvare i dati nelle rispettive variabili locali. Infine, nel file *input.txt*, essendo indicata anche la durata del turno di lavoro, viene impostato il periodo di osservazione (variabili *start_date* e *end_date*) in cui lo script deve prendere i dati dal database ed eseguire i calcoli per gli indici di prestazione.

Viene di seguito fornita una spiegazione del diagramma di flusso sotto forma di pseudo-codice, è possibile visionare il codice completo in Appendice A.

```
1 def main():
2     while 1:
3         chiamata al database
4         creazione delle tabelle csv
5         inserimento percorso del file di input
6         registrazione variabili dal file di input attraverso
              funzione specifica
```

```
7         creazione variabili per periodo di osservazione
8         ...
```

Questa sezione appena descritta contiene diverse funzioni implementate separatamente, la prima che viene eseguita è la chiamata al database della Festo.

```
1 def call_to_database():
2     inserimento percorso del database
3     avvio della connessione
4     creazione oggetto cursore per le query
5     esecuzione delle query
6     ordinamento dati in base alla data
7     salvataggio su file csv tramite apposita funzione
8     chiusura della connessione
```

Il salvataggio dei dati nelle tabelle *.csv* avviene nel seguente modo.

```
1 def tabCsv(titolo, results):
2     creazione o apertura 'titolo' in scrittura tramite funzione
          open()
3     creazione oggetto writer
4     separazione dati da ',' e terminazione riga da '\n'
5     scrittura dati da lista 'results'
```

Successivamente viene invocata la funzione per il caricamento delle impostazioni iniziali da file di testo esterno, se prima è stata eseguita una scrittura su file ora avviene una lettura.

```
1 def load_settings(path_file):
2     lettura dati da 'path_file'
3     suddivisione contenuto in riga per riga
4     conversione valori in interi
5     associazione dati letti a nuove variabili
```

Infine, viene eseguita la funzione per l'impostazione della data di inizio e fine osservazione presente nel modulo *date_new.py*. Questo periodo temporale verrà poi utilizzato nelle tabelle *.csv* per filtrare i dati ricavati dal database e prendere solo quelli che rientrano nel periodo impostato, per poi eseguire i calcoli e ricavare gli indicatori chiave di prestazione.

```
1 def caricamento_date(turno):
2     start_date = tempo attuale - il 'turno' di lavoro impostato
          dal file di 'input'
3     end_date = tempo attuale
```

Finita la fase di inizializzazione delle impostazioni iniziali, dopo aver acquisito tutte le informazioni necessarie all'adempimento dello scopo di questo elaborato, inizia la fase di elaborazione dei dati.

5.1.2. Manipolazione dati da tabella tblFinOrderPos

In Figura 5.5 viene mostrato l'avanzamento del codice nelle fasi che compongono il blocco di elaborazione dei dati dalla tabella tblFinOrderPos.

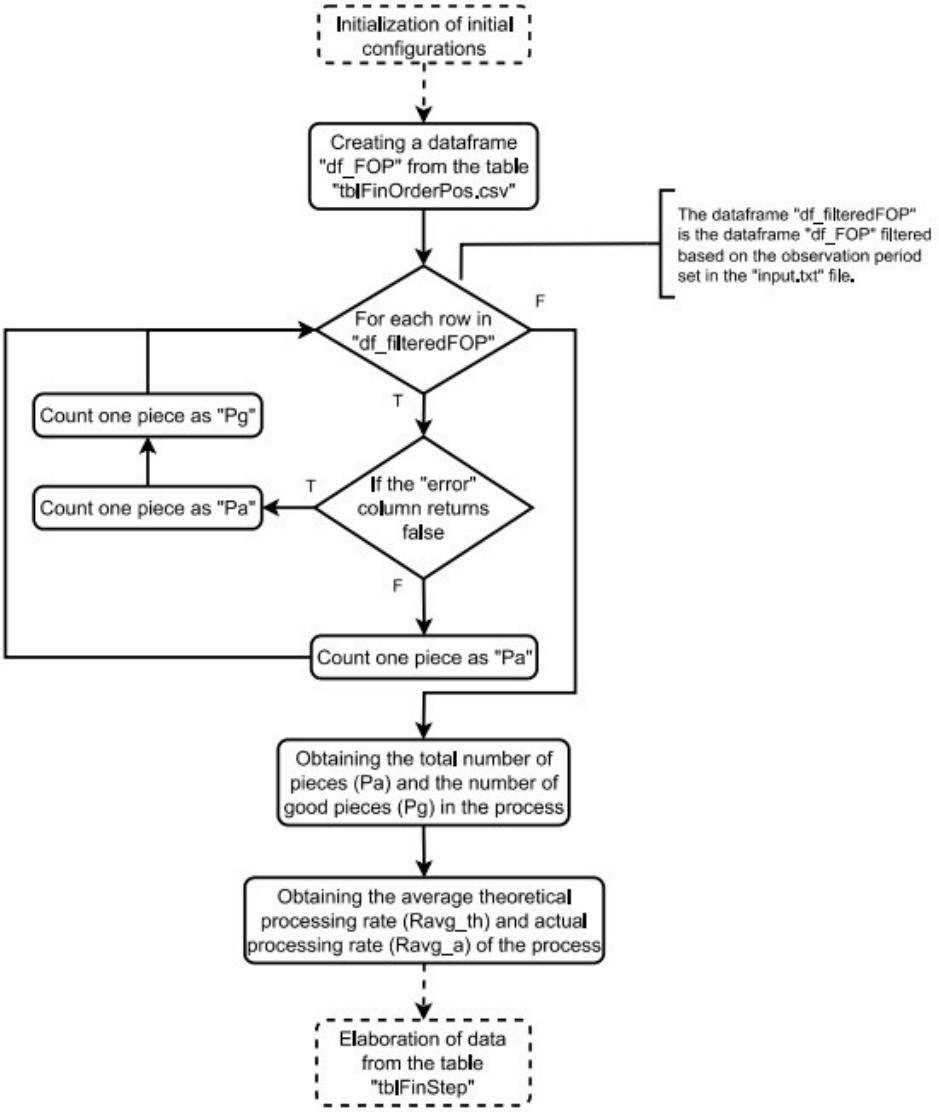


Figura 5.5.: Diagramma di flusso del blocco "tblFinOrderPos".

La prima fase è la creazione del dataframe *df_FOP* (grazie alla libreria *pandas*) utilizzando i dati della tabella *tblFinOrderPos.csv*, quest'ultima creata dopo la lettura dal database della Festo. L'utilizzo del dataframe è fondamentale, in quanto permette di effettuare più facilmente le lavorazioni sui dati ottenuti per eseguire le operazioni necessarie al calcolo degli indicatori chiave di prestazione.

Da qui in avanti il codice è inserito in un blocco *try-except* per gestire situazioni di errore nel caso in cui il periodo di osservazione inserito, in cui prelevare i dati, sia errato. Infatti, se viene inserito un intervallo temporale nel quale il database non ha

dati al suo interno: lo script ritorna il messaggio di errore *"Database vuoto"*, rimane in attesa per un certo periodo di tempo e poi riesegue la chiamata.

```
1  def main():
2      while 1:
3          ...
4          creazione dataframe df_FOP
5          assegnamento nomi colonne
6          conversione date stringa in oggetti datetime
7          creazione dataframe df_filteredFOP secondo il periodo
               di osservazione impostato
8          for riga in df_filteredFOP[colonna 'Error']:
9              if valore riga == false:
10                 contatore pezzi giusti Pg
11             contatore pezzi totali Pa
12         dataframe df_TimePlanedFOP = colonna fine tempo
               pianificato - colonna inizio tempo pianificato
13         dataframe df_TimeRealFOP = colonna fine tempo reale -
               colonna inizio tempo reale
14         ricavo tasso medio di elaborazione teorico e reale del
               processo
15         ...
```

Il ricavo del tasso medio di elaborazione teorico $R_{avg}^{(th)}$ e del tasso medio di elaborazione reale $R_{avg}^{(a)}$ del processo avviene in seguito alla creazione di due dataframe, *df_TimePlanedFOP* e *df_TimeRealFOP*, essi sono composti da n righe e da una sola colonna, successivamente: per $R_{avg}^{(th)}$ viene fatta la somma di tutti i tempi all'interno del rispettivo dataframe *df_TimePlanedFOP* diviso il numero di righe, per $R_{avg}^{(a)}$ viene fatta la somma di tutti i tempi all'interno del rispettivo dataframe *df_TimeRealFOP* diviso il numero di righe.

5.1.3. Manipolazione dati da tabella tblFinStep

Dopo aver ricavato le informazioni necessarie dalla tabella *tblFinOrderPos.csv*, si è passati all'analisi della tabella *tblFinStep.csv*. In Figura 5.6 è mostrato l'avanzamento del codice nelle successive fasi.

Figura 5.6.: Diagramma di flusso del blocco "tblFinStep".

Come nel paragrafo precedente, anche qui è stato necessario l'uso dei dataframe per un più semplice utilizzo delle informazioni, i dati di interesse sono tutti ricavati attraverso un unico ciclo *for* e l'utilizzo di molteplici istruzioni condizionali *if*.

```
1  def main():
2      while 1:
3          ...
4          creazione dataframe df_FS
5          assegnamento nomi colonne
6          conversione date stringa in oggetti datetime
7          creazione dataframe df_filteredFS secondo il periodo di
               osservazione impostato
8          for riga in df_filteredFS:
9              if valore riga[colonna 'ResourceID'] == stazione
                   PickByLight:
10                 incremento variabile temporanea teorica PBL
11                 incremento variabile temporanea reale PBL
12             if valore riga[colonna 'ResourceID'] == stazione
                   Rework and not valore nullo:
13                 incremento variabile temporanea teorica RW
14                 incremento variabile temporanea reale RW
```

```
15              if valore riga[colonna 'ResourceID'] == stazione AS
                   /RS and not valore nullo:
16                  incremento variabile temporanea teorica AS/RS
17                  incremento variabile temporanea reale AS/RS
18              if valore riga[colonna 'ResourceID'] == stazione
                   Camera 1 and not valore nullo:
19                  incremento variabile temporanea teorica Camera
                       1
20                  incremento variabile temporanea reale Camera 1
21              if valore riga[colonna 'ResourceID'] == stazione
                   Camera 2 and not valore nullo:
22                  incremento variabile temporanea teorica Camera
                       2
23                  incremento variabile temporanea reale Camera 2
24              if valore riga[colonna 'ResourceID'] == stazione
                   Magazzino Gusci and not valore nullo:
25                  incremento variabile temporanea teorica
                       Magazzino Gusci
26                  incremento variabile temporanea reale Magazzino
                        Gusci
27              if valore riga[colonna 'ResourceID'] == stazione
                   Camera 1 and valore riga[colonna 'ErrorRetVal']
                   == codice errore:
28                  incremento variabile contatore per PickByLight
29              if valore riga[colonna 'ResourceID'] == stazione
                   Camera 2 and valore riga[colonna 'ErrorRetVal']
                   == codice errore:
30                  incremento variabile contatore per Rework
31              if valore riga[colonna 'ResourceID'] == stazione
                   Rework and valore riga[colonna 'ErrorRetVal'] ==
                    codice errore:
32                  incremento variabile contatore per Magazzino
                       Gusci
33          ricavo tassi medi di elaborazione teorici e reali di
               ciascuna stazione
34          ricavo pezzi totali e giusti di ciascuna stazione
35          ricavo perdite MWL e MDL per stazioni manuali
36          ...
```

A questo punto del codice, sono stati ricavati tutti i dati necessari al calcolo dei parametri di *Efficienza* e *Qualità* relativi all'indice OEE per l'intera macchina CPFactory e per le singole stazioni. Sono state inoltre ricavate, come descritto in Tabella 3.4, le perdite MWL e MDL, rispettivamente legate alla macchina e quelle causate dalla gestione, per il calcolo dell'indice ROLE (attraverso l'Equazione 3.6) nelle stazioni manuali *Pick By Light* e *Rework* (maggiori dettagli al Capitolo 3, sezione ROLE). Nel paragrafo successivo verranno ricavati tutti i parametri necessari al calcolo della *Disponibilità* della macchina.

5.1.4. Manipolazione dati da tabella tblMachineReport

Di seguito (Figura 5.7) è mostrato il diagramma di flusso del blocco per l'elaborazione dei dati dalla tabella tblMachineReport, l'ultima tabella che viene consultata per l'ottenimento di tutti i dati necessari allo scopo finale.

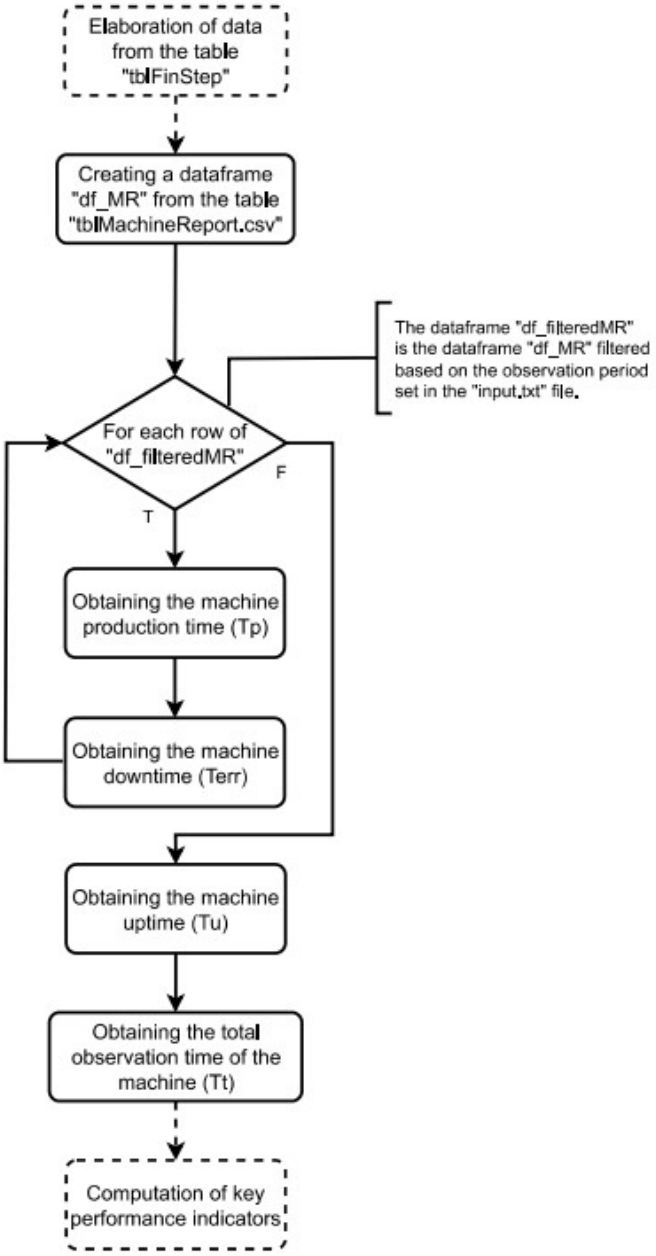


Figura 5.7.: Diagramma di flusso del blocco "tblMachineReport".

Come già descritto in precedenza, in questa tabella sono presenti i dati necessari al calcolo del parametro di *Disponibilità* della CPFactory presente nei laboratori i-Labs.
Viene dunque mostrato lo pseudo-codice di questa sezione di codice.

```
def main():
    while 1:
        ...
        creazione dataframe df_MR
        assegnamento nomi colonne
        conversione date stringa in oggetti datetime
        creazione dataframe df_filteredMR secondo il periodo di
            osservazione impostato
        for riga in df_filteredMR:
            if valore riga[colonna 'AutomaticMode'] == true:
                ricavo tempo di produzione Tp
            if valore riga[colonna 'ManualMode'] == true or
               valore riga[colonna 'Reset'] == true or valore
               riga[colonna 'ErrorL0'] == true or valore riga[
               colonna 'ErrorL1'] == true or valore riga[
               colonna 'ErrorL2'] == true:
                ricavo tempo di errore Terr
        ricavo tempo totale Tt = end_date - start_date
        ricavo tempo di uptime Tu = Tt - Terr
        ...
```

Una considerazione da fare è che, per il valore della *Disponibilità (A)* delle singole stazioni (di conseguenza per il calcolo OEE delle singole stazioni), è stato utilizzato lo stesso valore della macchina generale in quanto i moduli della CPFactory lavorano in maniera coordinata, se un modulo va in errore allora tutta la macchina si ferma.

5.1.5. Calcolo indicatori chiave di prestazione (KPI)

In questa sezione sono stati eseguiti i calcoli finali di tutti gli indicatori chiave di prestazione, necessari per avere un quadro generale sulla produttività della CPFactory. In Figura 5.8 è possibile seguire passo passo l'ordine con cui vengono ricavati tali valori. Inoltre è stata realizzata una interfaccia grafica su web che permette la visualizzazione ordinata di questi dati attraverso dei grafici, come utilizzarla ed il codice usato per implementarla sono descritti rispettivamente al Capitolo 8 e in Appendice B.

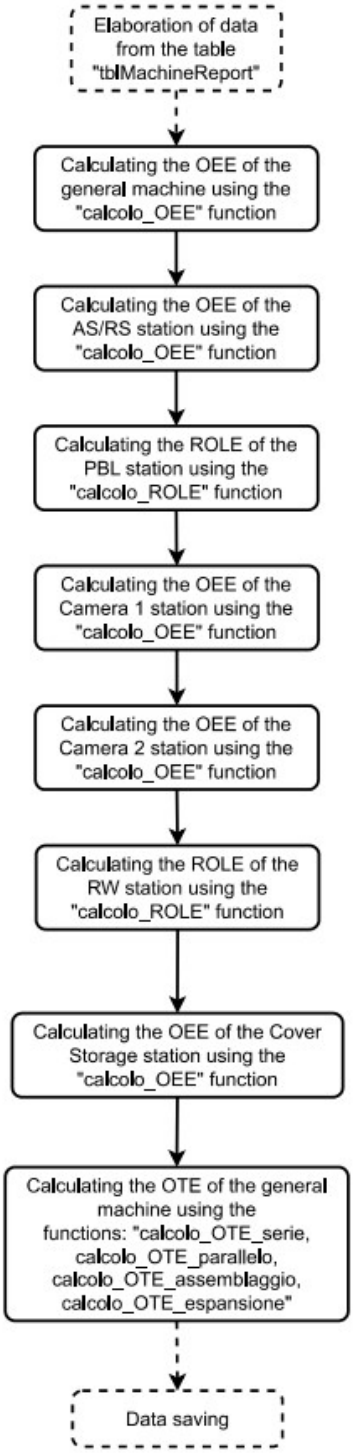


Figura 5.8.: Diagramma di flusso del blocco per il calcolo indicatori chiave di prestazione.

Di seguito viene esposto lo pseudo-codice della sezione dedicata al calcolo degli indicatori chiave di prestazione.

```
def main():
    while 1:
        ...
        calcolo Disponibilita CPF
        calcolo Efficienza CPF
        calcolo Qualita CPF
        calcolo OEE generale CPF
        calcolo OEE stazione AS/RS
        calcolo ROLE stazione Pick By Light
        calcolo OEE stazione Camera 1
        calcolo ROLE stazione Rework
        calcolo OEE stazione Camera 2
        calcolo OEE stazione Magazzino Gusci
        calcolo OTE sottosistema in serie 1
        calcolo OTE sottosistema in serie 2
        calcolo OTE sottosistema in parallelo 1
        calcolo OTE generale CPF
        ...
```

Per il calcolo OEE(3.2), ROLE(3.6) e OTE(3.11 e 3.12) nelle varie configurazioni sono state utilizzate le rispettive funzioni, seguendo le formule descritte nel Capitolo 3: *calcolo_OEE*, *calcolo_ROLE*, *calcolo_OTE_serie*, *calcolo_OTE_parallelo*. Inoltre per l'indice ROLE è stato necessario ricavare i tre fattori di perdita con cui viene calcolato, dunque per calcolarlo sono state usate le funzioni: *calcolo_OWE*, *calcolo_ME*, *calcolo_WA*.

5.1.6. Salvataggio dati

Dopo aver ricavato tutti gli indicatori di prestazione utili alle conclusioni di questo elaborato, i dati vengono salvati e archiviati in un database locale chiamato *"MyDatabase"* per mantenere uno storico della produttività della CPFactory, il diagramma in Figura 5.9 riassume i passaggi principali eseguiti in questa sezione di codice.

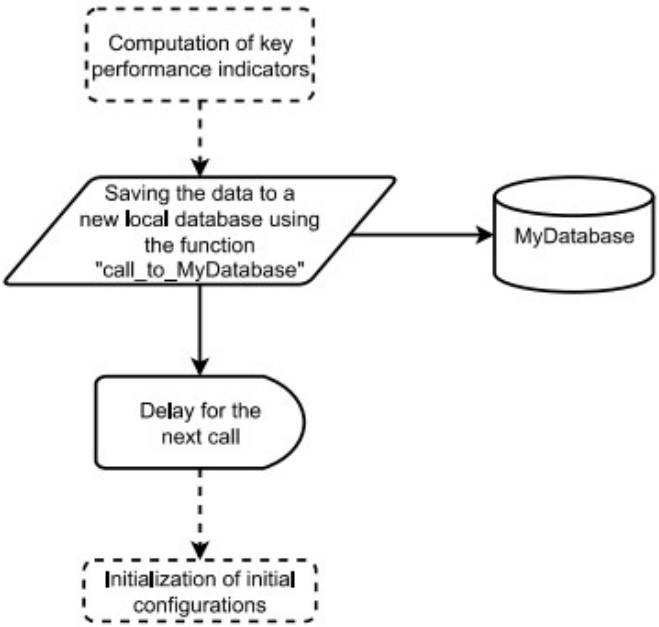


Figura 5.9.: Diagramma di flusso del blocco per il salvataggio dei dati".

In seguito alla chiamata a *MyDatabase* è presente un *delay* (ritardo) che ferma il codice per un determinato periodo di tempo, successivamente vengono ricaricate le variabili *start_date* e *end_date* con le date aggiornate e rieseguito il codice dall'inizio.

La rappresentazione dello pseudo-codice in questo caso è molto semplice.

```
def main():
    while 1:
        ...
        chiamata al mio database
        delay
        reload date per periodo di osservazione
```

La funzione *"call_to_MyDatabase(elenco parametri)"* è così strutturata.

```
def call_to_MyDatabase(elenco parametri):
    inserimento credenziali database
    specifica nome tabella
    avvio della connessione
    creazione oggetto cursore per le query
    creazione della tabella se non esiste
    inserimento dei dati al suo interno
    commit dei cambiamenti
    chiusura della connessione
```

Capitolo 6.

Rappresentazione della CPFactory tramite sottosistemi

Come anticipato nel Capitolo 3, l'indice OTE (Overall Throughput Effectiveness) è utilizzato per misurare le prestazioni a livello di fabbrica.
Lo scopo di questo capitolo è di confrontare i risultati dell'indice OEE complessivo della CPFactory, calcolato direttamente dal software appena descritto, con un modello che considera l'intera linea di produzione come una serie di sottosistemi interconnessi, di cui si vuol monitorare singolarmente l'efficienza per poi risalire ad un indice complessivo di linea. Infine mettere a confronto i risultati ottenuti in modo che si possa diversificare l'effetto dell'operatore da quello delle macchine per poi aggregare tutto in un indice generale da cui poter dedurre conclusioni più puntuali e precise dei fenomeni accaduti.

Dal momento che l'OTE viene applicato a sottosistemi, è necessario creare una configurazione adeguata della CPFactory adoperando lo schema a blocchi già introdotto nel Capitolo 1 e di seguito riportato per una maggiore comprensione.

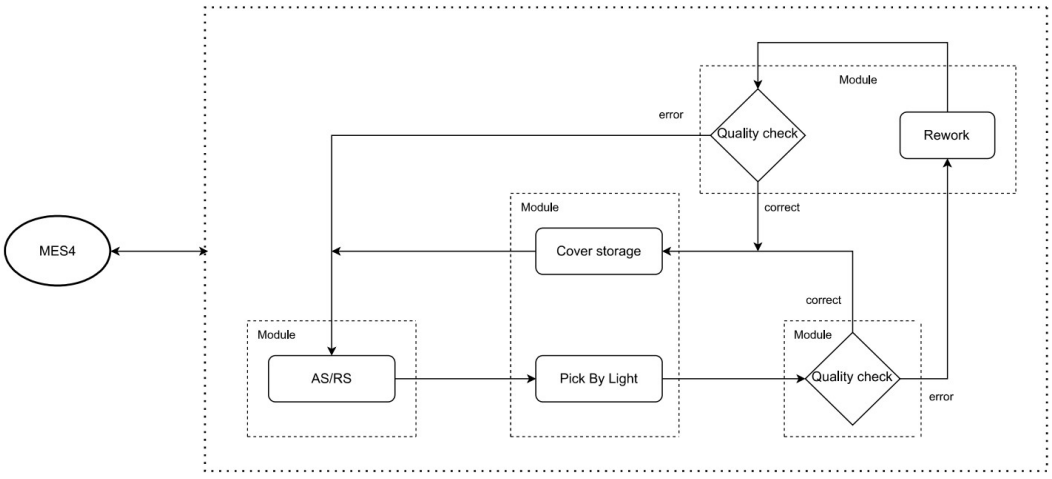


Figura 6.1.: Schema a blocchi della CPFactory.

Mentre in Figura 6.2 è mostrato lo schema generale utilizzato per ricavare l'indice OTE, tale schema è quindi derivato da quello in Figura 6.1 ed è stato riprodotto

in termini di interconnessioni di sistemi ricadenti nelle 4 tipologie di connessione previste per l'indice OTE.

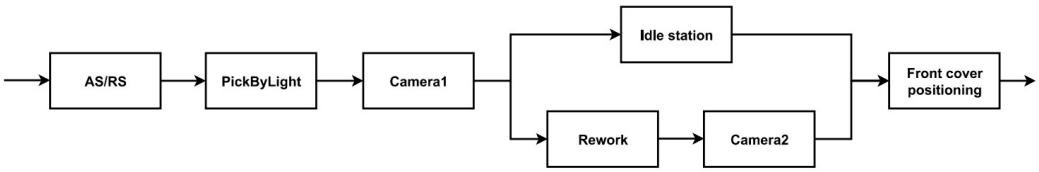


Figura 6.2.: Blocchi della CP_F.

Lo schema appena raffigurato descrive l'avanzamento del processo nelle rispettive fasi, subito dopo la Camera 1 si ha uno sdoppiamento della linea principale in cui:

- se il prodotto è corretto, prosegue la sua strada verso il Posizionamento dei gusci anteriori, nel mezzo però è stata aggiunta la stazione *Idle station*, la quale è una stazione ideale "fantasma", ovvero che non esiste nella realtà, ed è inserita per raffigurare il semplice trasporto del prodotto dalla Camera 1 al Posizionamento gusci anteriori nel caso non ci siano difetti;
- se il prodotto è difettoso, avviene la situazione in cui viene guidato sulla linea di Rework e Camera 2 per un ulteriore controllo di qualità, al termine della rilavorazione si ricongiunge alla linea principale per il posizionamento del guscio anteriore.

Tenendo quindi conto delle quattro configurazioni illustrate nel Capitolo 3, ovvero: serie, parallelo, assemblaggio ed espansione; è susseguita l'analisi delle possibili configurazioni da combinare ai fini di trovare un OTE generale che descriva le prestazioni dell'intera CPFactory.

6.1. Analisi delle possibili configurazioni

La scelta della configurazione di interconnessione adeguata tra sottosistemi per il calcolo dell'OTE generale non è stata immediata. Sono state realizzate diverse configurazioni prima di arrivare a quella corretta. Il primo passo dunque è stato quello di ricavare l'OTE dei rispettivi sottosistemi adottati. In Figura 6.3 è mostrata la tabella utilizzata per lo sviluppo degli OTE nelle diverse configurazioni.

	Start Date (dd/MM/yy - HH:mm) →	23/06/23 - 10:53	24/05/23 - 11:50	24/05/23 - 11:18	17/05/23 - 11:37	17/05/23 - 11:03	17/05/23 - 10:07	12/05/23 - 14:11
	End Date (dd/MM/yy - HH:mm) →	23/06/23 - 11:03	24/05/23 - 12:20	24/05/23 - 11:48	17/05/23 - 12:07	17/05/23 - 11:33	17/05/23 - 10:37	12/05/23 - 14:41
	T. OBSERVATION	10:30 min	30 min	30 min	30 min	30 min	30 min	30 min
	T. PRODUCTION	8:43 min	9:03 min	12:20 min	19:44 min	16:32 min	24:41 min	9 min
KPI ↓	**OEE_generalCPF**	81%	28%	51%	22%	42%	74%	30%
	(%)	**Test 1**	**Test 2**	**Test 3**	**Test 4**	**Test 5**	**Test 6**	**Test 7**
	ROLE_PBL	89	71	61	19	59	85	43
	ROLE_RW	98	97	73	96	42	83	50
	OEE_AS/RS	73	31	47	68	53	86	30
	OEE_CAM1	72	25	39	62	39	71	28
	OEE_CAM2	74	27	41	64	39	72	29
	OEE_COVER	75	29	46	65	54	85	29
Rth ↓	**(sec)**							
	wpTimePlanedPBL	21	21	21	21	21	21	21
	wpTimePlanedRW	21	23	21	21	20	20	21
	avgR1_planed	27	29	28	29	27	27	28
	avgR3_planed	26	26	26	26	26	26	26
	avgR5_planed	22	22	22	22	22	22	22
	avgR6_planed	80	80	80	80	80	80	80

Figura 6.3.: Indici di prestazione delle singole stazioni.

Le prime due righe indicano le date di inizio e fine osservazione in cui sono stati prelevati i dati, la terza riga indica il tempo totale di osservazione (differenza tra data di fine e inizio osservazione), la quarta riga indica il tempo di produzione effettivo dei rispettivi test. In seguito sono raffigurati tutti i valori di OEE e ROLE delle singole stazioni, espressi in percentuale (%), e tutti i valori dei tassi medi di elaborazione teorici espressi in secondi (sec). La scelta di raffigurare questi valori è data dal fatto che sono i principali parametri utilizzati nelle formule per il calcolo dell'indice OTE nelle quattro configurazioni principali.

Average theoretical processing rate
avgR1_planed = AS/RS station
wpTimePlanedPBL = PickByLight station
avgR3_planed = Camera 1 station
wpTimePlanedRW = Rework station
avgR5_planed = Camera 2 station
avgR6_planed = Cover Storage station

Figura 6.4.: Didascalia variabili.

In Figura 6.4 è mostrata la didascalia delle variabili dei tassi medi di elaborazione teorici associate alle rispettive stazioni della CPFactory. E' doveroso aggiungere che i tassi medi di elaborazione, oltre al tempo di lavorazione della stazione, sono composti anche dal tempo di trasporto che inizia dalla stazione presa in considerazione e termina all'arrivo nella successiva. In teoria, i tempi raffigurati in Figura 6.3, essendo di "elaborazione teorica", non dovrebbero variare tra le diverse prove; in pratica, alcuni valori variano di qualche secondo a causa di piccoli aggiustamenti temporali eseguiti sul MES nel laboratorio tra una prova e l'altra (tali dati sono prelevati proprio dal MES). Queste piccole variazioni non hanno comportato cambiamenti rilevanti ai fini dell'elaborato.

Nel seguito verranno mostrati 3 esempi di tentativi di interconnettere i sottosistemi senza seguire linee guida specifiche dedotte da riscontri pratici determinati dalla

fisica del processo. Si mostrerà quindi come questa modalità di procedere porta a 3 soluzioni non coerenti con i dati reali del processo stesso, permettendo però di elaborare delle linee guida che permetteranno di identificare l'interconnessione corretta idonea a modellare il processo stesso in modo coerente con i risultati fisici.

Un primo tentativo di interconnessione dello schema 6.2 in sottosistemi descritti dalle quattro configurazioni principali viene mostrata in Figura 6.5. La *Serie 1* è composta dalle stazioni del modulo AS/RS e Pick By Light. La *Serie 2* invece è creata dall'unione delle stazioni Rework e Camera 2. Successivamente viene utilizzata la configurazione di *Espansione 1* con le stazioni di: Camera 1, Stazione ideale e Serie 2. Infine la configurazione *Assemblaggio 1* comprende le stazioni di: Posizionamento gusci anteriori, Stazione ideale e Serie 2.

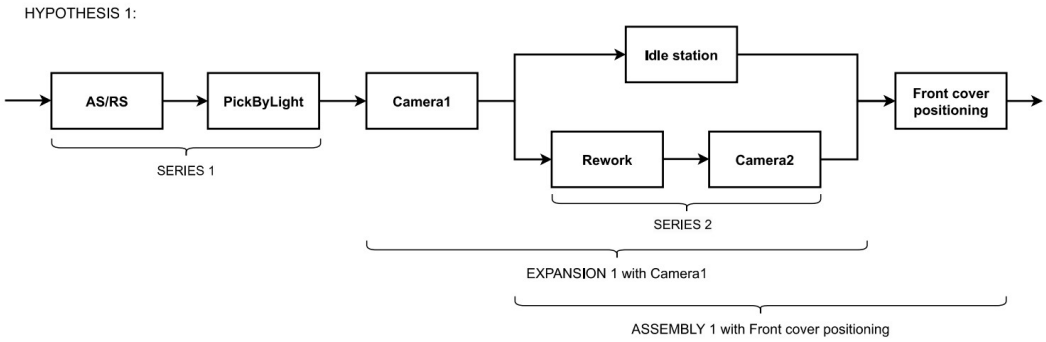


Figura 6.5.: Interconnessione ipotesi 1.

Dopo aver suddiviso lo schema generico in sottosistemi riconducibili alle quattro configurazioni principali, seguendo le formule (3.11, 3.13, 3.14) definite nel Capitolo 3, sono stati sviluppati i calcoli dei rispettivi sottosistemi per ciascun test eseguito. In prima analisi, risultati con valori oltre il 100% non saranno ritenuti accettabili.

Come è possibile osservare in Figura 6.6, i dati relativi a *OTE_SERIES1* e *OTE_SERIES2* ritornano valori apparentemente accettabili in un contesto di valutazione per l'OTE generale; mentre i dati presenti nelle righe di *OTE_EXP1*, riferito al sottosistema di Espansione 1, e *OTE_ASS1*, riferito al sottosistema di Assemblaggio 1, non ritornano valori accettabili per il proseguimento della determinazione dell'indice OTE generale per la CPFactory con questa interconnessione, dal momento che per più prove elencate i loro valori superano il 100%.

HYPOTHESIS 1 --> EXPANSION1 with CAM1

	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7
OTE_SERIES1	89%	43%	61%	19%	59%	85%	42%
OTE_SERIES2	77%	27%	44%	67%	42%	77%	30%
OTE_EXP1	114%	74%	86%	105%	84%	113%	75%
OTE_ASS1	232%	91%	142%	200%	168%	262%	91%

Figura 6.6.: OTE sottosistemi in ipotesi 1.

Un secondo tentativo di interconnessione dello schema generale nei principali sottosistemi è raffigurato in Figura 6.7. La *Serie 1* in questa ipotesi è costituita dalle stazioni del modulo AS/RS, Pick By Light e Camera 1. La *Serie 2*, come nel caso precedente, è composta dalla stazione Rework e Camera 2. In questa versione la configurazione di *Espansione 1* comprende l'intera Serie 1, la Serie 2 e la Stazione ideale. L'*Assemblaggio 1*, come la Serie 2, è identico alla precedente ipotesi.

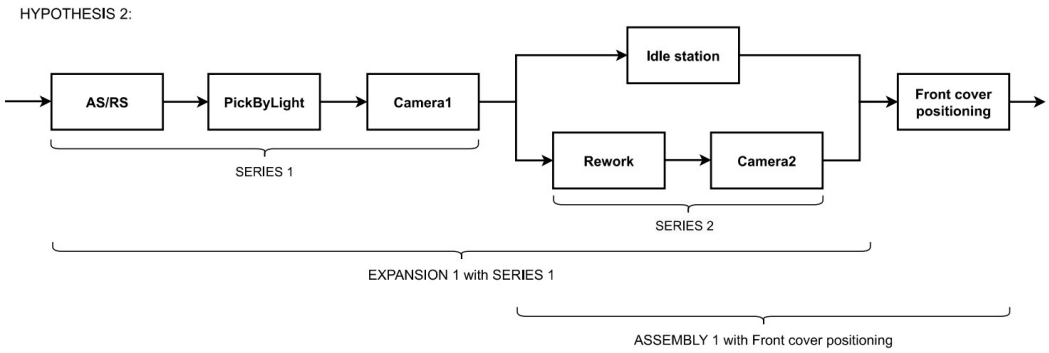


Figura 6.7.: Interconnessione ipotesi 2.

Come è possibile osservare in Figura 6.8, i valori della Serie 2 e di Assemblaggio 1 rimangono invariati dal momento che si ricavano allo stesso modo della ipotesi precedente. In questa seconda interconnessione proposta però si hanno valori di *OTE_EXP1* più contenuti, tuttavia i dati nella riga di *OTE_ASS1* sono sempre fuori range ammissibile (oltre il 100%), dunque anche per questa ipotesi non è possibile avanzare calcoli ulteriori per il ricavo dell'indice OTE generale.

HYPOTHESIS 2 --> EXPANSION1 with SERIES 1

	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7
OTE_SERIES1	89%	31%	49%	19%	48%	85%	35%
OTE_SERIES2	77%	27%	44%	67%	42%	77%	30%
OTE_EXP1	85%	46%	54%	72%	54%	80%	66%
OTE_ASS1	232%	91%	142%	200%	168%	262%	91%

Figura 6.8.: OTE sottosistemi in ipotesi 2.

Un terzo tentativo di interconnessione è stato quello di escludere dall'equazione il sottosistema di *Assemblaggio 1* come raffigurato in Figura 6.10. Dal momento che i blocchi di Serie 2 e Stazione ideale sono considerati già nel sottosistema di Espansione 1, è possibile ipotizzare un'interconnessione di questo tipo. Non è considerato un errore lasciare la stazione di Posizionamento gusci anteriori senza inserirla in un sottosistema in quanto, data la natura delle formule per l'OTE, è sufficiente considerare il modulo Posizionamento gusci anteriori come un sottosistema a sé e di conseguenza utilizzare il suo indice OEE come indice OTE nelle successive fasi.

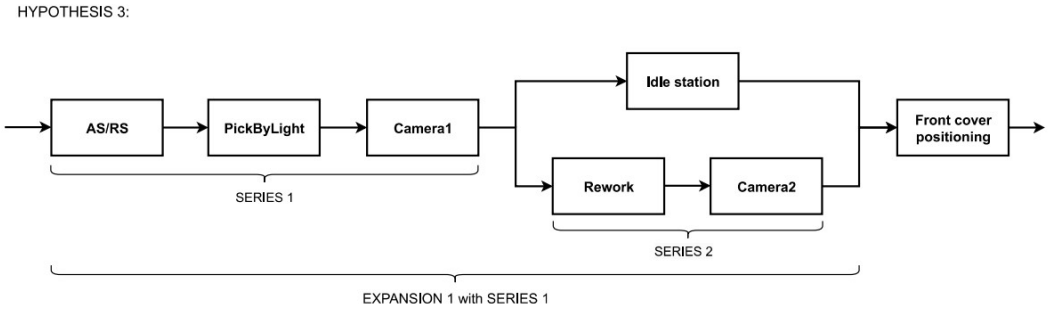


Figura 6.9.: Interconnessione ipotesi 3.

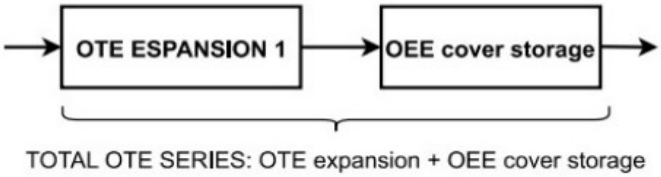
I valori ricavati (Figura 6.10) attraverso le formule dei rispettivi sottosistemi sembrano essere in linea con i dati presenti in Figura 6.3, ovvero i parametri dei relativi sottosistemi sono in linea con gli indici di prestazione delle singole stazioni presenti in tali sottosistemi, dunque si è deciso di approfondire lo studio per calcolare l'indice OTE generale della CPFactory. L'unico metodo di verifica per la correttezza dei sottosistemi è infatti mettere a confronto l'OTE generale della macchina con l'indice OEE generale, inserito nella tabella in Figura 6.3, per le rispettive prove di test.

HYPOTHESIS 3 --> without ASSEMBLY1

	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7
OTE_SERIES1	89%	31%	49%	19%	48%	85%	35%
OTE_SERIES2	77%	27%	44%	67%	42%	77%	30%
OTE_EXP1	85%	46%	54%	72%	54%	80%	66%

Figura 6.10.: OTE sottosistemi in ipotesi 3.

Come già anticipato, il ricavo dell'OTE generale avviene utilizzando sempre le configurazioni principali e le formule ad esse associate in modo ricorsivo. Nella configurazione raffigurata in Figura 6.11 si è scelto di mettere in *serie* il sottosistema Espansione 1 con la stazione Posizionamento gusci anteriori, la formula utilizzata per l'indice generale è dunque la 3.11 descritta al Capitolo 3. Successivamente al grafico è mostrata la tabella contenente i valori di OTE generale della CPFactory per ogni test preso in esame.



	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7
OTE_TOTAL	74%	29%	42%	62%	54%	83%	29%

Figura 6.11.: OTE totale in ipotesi 3.

Utilizzando l'ipotesi 3 dunque è possibile ricavare l'indice OTE generale della macchina, è ora necessario verificare la correttezza di questi valori. Mettendoli a confronto con il valore di OEE generale (presente in Figura 6.3, "OEE_generalCPF") si può notare come nella maggior parte delle prove il valore di OTE_TOTAL non coincida (non è necessario che sia esattamente uguale a *OEE_generalCPF*, ma che almeno si aggiri nei suoi dintorni). Si può concludere che nemmeno questa ipotesi di interconnessione è corretta.

Analizzando meglio la situazione reale, ovvero di come effettivamente avviene il processo produttivo nella CPFactory, l'utilizzo delle configurazioni di espansione ed assemblaggio non sono corretti. Infatti la configurazione di assemblaggio ha, come operazione fisica, il fatto che i pezzi provenienti da più stazioni operative vengano assemblati per formare un pezzo unico; questo non accade nella CPFactory dal momento che il pezzo viene indirizzato su una linea o in quella sottostante (riferendosi allo schema generico). Stessa considerazione vale per la configurazione di espansione: il suo significato fisico è la frammentazione di un pezzo singolo in più pezzi separati su più linee di produzione a seguire. Questa osservazione è di fondamentale importanza in quanto permette di sfruttare un'altra configurazione principale non ancora utilizzata, ovvero il sottosistema in parallelo, e di dedurre delle linee guida per lo sviluppo delle interconnessioni. Queste vengono riassunte nel paragrafo successivo.

6.1.1. Linee guida per lo sviluppo di interconnessioni

Prima di proseguire con la configurazione utilizzata per lo sviluppo dell'elaborato, l'analisi di queste configurazioni appena descritte ha permesso la stesura di alcune linee guida da seguire per la realizzazione di interconnessioni tra sottosistemi in un contesto più generale:

1. Considerare i sottosistemi una sola volta nelle combinazioni dei raggruppamenti.
2. L'operazione di Espansione è associata alla separazione dei pezzi da uno a molti.
3. L'operazione di Assemblaggio è associata all'accorpamento dei pezzi da molti a uno.
4. L'operazione di Parallelo è associata alla selezione/indirizzamento del pezzo da uno a uno (ovvero se nel sottosistema di parallelo è presente in ingresso un oggetto con peso unitario, all'uscita del sottosistema in parallelo è nuovamente presente un oggetto con peso unitario).

6.2. Configurazione di interconnessione corretta

La configurazione finale adottata per ricavare l'indice OTE della CPFactory è quella mostrata in Figura 6.12. Questa interconnessione utilizza la *Serie 1* composta

dalle stazioni del modulo AS/RS, Pick By Light e Camera 1. La *Serie 2* è strutturata come le precedenti prove, ovvero unisce le stazioni di Rework e Camera 2. Inoltre viene proposto l'utilizzo della configurazione in *Parallelo* tra la Serie 2 e la Stazione ideale.

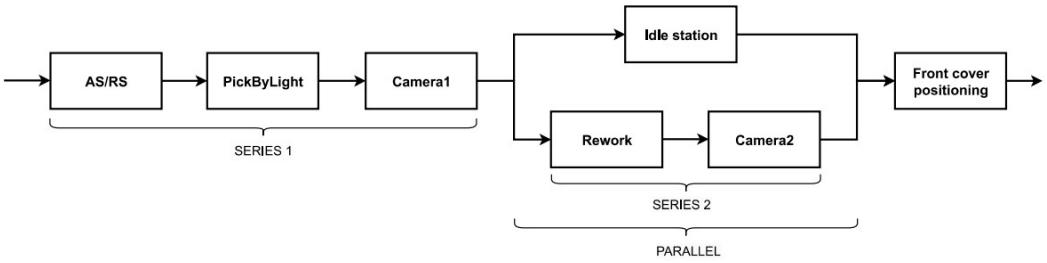


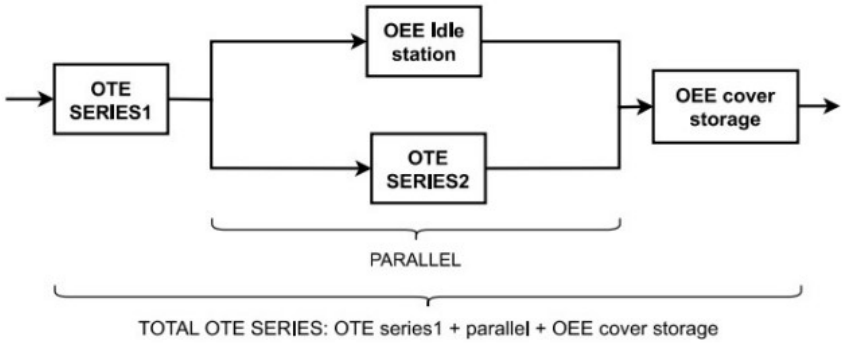
Figura 6.12.: Interconnessione adottata (ipotesi 4).

La riga di *OTE_PARALLEL* in Figura 6.13 è ottenuta utilizzando la formula 3.12 espressa nel Capitolo 3. Anche in questo caso i valori ottenuti dalla composizione dei sottosistemi appena introdotti sembrano essere in linea con l'andamento generale nelle rispettive prove di test e non superano il 100%. Dunque è possibile continuare lo studio per il ricavo dell'OTE generale seguendo lo schema di quest'ultima interconnessione proposta.

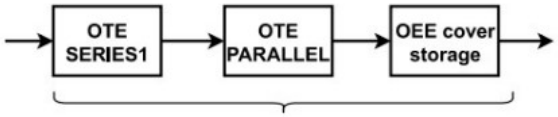
	USED CONFIGURATION --> without EXPANSION1 and ASSEMBLY1						
	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7
OTE_SERIES1	89%	31%	49%	19%	48%	85%	35%
OTE_SERIES2	77%	27%	44%	67%	42%	77%	30%
OTE_PARALLEL	86%	54%	65%	79%	64%	86%	56%

Figura 6.13.: OTE sottosistemi nella configurazione adottata (ipotesi 4).

Lo schema in Figura 6.12 può dunque essere riscritto nel seguente modo (Figura 6.14), continuando a raggruppare i sottosistemi proposti si giunge alla configurazione finale. Tale configurazione è il calcolo della *serie* tra i sottosistemi di Serie 1, Parallelo e Posizionamento gusci anteriori.



OTE FINAL CONFIGURATION:



TOTAL OTE SERIES: OTE series1 + parallel + OEE cover storage

	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7
OTE_TOTAL	84%	32%	53%	21%	44%	78%	34%

Figura 6.14.: OTE totale della configurazione adottata (ipotesi 4).

Come è possibile osservare dalla tabella dell'OTE generale (Figura 6.14), utilizzando questa interconnessione i valori di OTE della macchina (Figura 6.14) sono in linea con quelli dell'OEE generale (Figura 6.3). Questa osservazione è stata fatta mettendo a confronto ciascun valore delle tabelle appena menzionate per ogni prova riportata. Nei casi in cui ho degli indici di prestazione bassi nelle singole stazioni (come nei Test 2, 4 e 7 di Figura 6.3), tali valori vanno a incidere maggiormente nel risultato dell'OTE.

In questo capitolo dunque, è stata trovata una configurazione di interconnessione tra i sottosistemi della CPFactory adeguata per il ricavo dell'indice di prestazione OTE a livello di fabbrica, utilizzando le configurazioni principali proposte nel Capitolo 3. Inoltre è corretto che i valori dell'indice OTE e quelli dell'indice OEE siano simili in quanto anche quest'ultimo descrive l'andamento generale della macchina, considerando però i tempi di elaborazione teorici e reali dell'intero processo di produzione dei prodotti.

Capitolo 7.

Prove di validazione finale dei risultati

Nel capitolo precedente è stata trovata una interconnessione valida tra sottosistemi per ricavare l'indice OTE complessivo a livello dell'intera CPFactory. In quest'ultima sezione dell'elaborato viene riportata una prova significativa avente una durata sufficientemente elevata e significativa per i tempi di produzione di un processo industriale ed un numero di lavorazioni altrettanto elevate sviluppata proprio per mettere in evidenza la relazione tra l'indice OEE e l'indice OTE della macchina generale.

La prova è stata svolta considerando un turno lavorativo di due ore, la disponibilità della CPFactory è stata tenuta volontariamente intorno al 75%, di conseguenza, su 120 minuti la macchina ha avuto fermi programmati durante la produzione per un totale di 30 minuti. La capacità produttiva della macchina a pieno regime, in un turno lavorativo di due ore, si aggira intorno ai 96 pezzi totali, però con una riduzione del tempo di attività della macchina si ha un calo della capacità produttiva a circa 72 pezzi. Per questa prova inoltre è stato scelto di rendere difettosi 4 di questi pezzi, dunque si avranno 68 pezzi corretti e 72 pezzi totali.

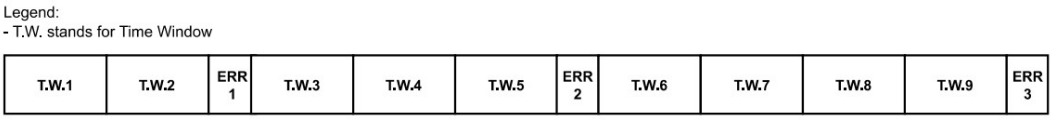


Figura 7.1.: Timeline della prova di validazione finale.

In Figura 7.1 è mostrata la linea temporale della prova avvenuta nelle due ore, ogni sezione è una finestra temporale in cui è stato eseguito un ciclo di produzione di 8 pezzi. Per questioni legate alla disponibilità numerica dei pallet nel modulo AS/RS (come descritto nel Capitolo 1), e per una maggiore efficienza nella gestione della prova da parte dell'operatore che esegue il test, la macchina è limitata ad una produzione di 8 pezzi al massimo per ogni finestra temporale, per poi essere svuotata dei prodotti finiti che sono stati immagazzinati e riportata allo stato di ready tra una finestra temporale e l'altra.

7.1. Descrizione delle Time Windows

All'interno del turno lavorativo di due ore sono state eseguite 9 lavorazioni. Queste 9 lavorazioni sono state anticipatamente programmate per avere situazioni diverse durante il loro ciclo produttivo, al fine di evidenziare successivamente le differenze tra di loro e come variano gli indici OTE e OEE generali della macchina al variare della finestra temporale.

La Time Window 1 è la prima lavorazione effettuata nell'arco temporale di due ore, ha una durata di 8 minuti e 37 secondi. La prova è stata svolta in un ambiente ideale, quindi nessun pezzo scartato e nessun ritardo nelle stazioni di lavorazione manuale.

La Time Window 2 è la seconda lavorazione effettuata nell'arco temporale di due ore, ha una durata di 8 minuti e 54 secondi. La prova è stata svolta in un ambiente ideale, quindi nessun pezzo scartato e nessun ritardo nelle stazioni di lavorazione manuale. Seppur entrambe le prime due lavorazioni siano senza ritardi, la seconda è risultata comunque più lenta, questo è dovuto al fatto che lungo la linea possono esserci ritardi non programmati nei tempi di trasporto tra una stazione e l'altra (introdotti dall'operatore umano per evitare situazioni di code).
Dopo le prime due lavorazioni è stato inserito un tempo di errore macchina (ERR 1) di circa 6 minuti.

La Time Window 3 è la terza lavorazione effettuata nell'arco temporale di due ore, questa prova introduce dei ritardi nella stazione manuale Pick By Light con dei tempi di lavorazione pari al *doppio* della normale esecuzione, infatti la prova complessiva ha una durata di 9 minuti e 38 secondi. Nessun pezzo prodotto è stato scartato.

La Time Window 4 è la quarta lavorazione effettuata nella linea temporale di due ore, questa prova introduce dei ritardi nella stazione manuale Pick By Light con dei tempi di lavorazione pari al *triplo* della normale esecuzione, infatti la prova complessiva ha una durata di 10 minuti e 50 secondi. Inoltre sono stati scartati due pezzi.

La Time Window 5 è la quinta lavorazione eseguita nell'arco temporale di due ore, questa prova introduce dei ritardi nella stazione manuale Rework con dei tempi di lavorazione pari al *doppio* della normale esecuzione, la prova complessiva ha una durata di 9 minuti e 02 secondi. Nessun pezzo è stato scartato durante la produzione. Tra la quinta e la sesta lavorazione è inserito un altro tempo di errore macchina (ERR 2) dalla durata di circa 6 minuti.

La Time Window 6 è la sesta lavorazione eseguita durante la linea temporale di due ore, questa prova introduce dei ritardi nella stazione manuale Rework con dei tempi di lavorazione pari al *triplo* della normale esecuzione, la prova complessiva ha una durata di 9 minuti e 31 secondi. Anche in questo caso nessun pezzo è stato scartato durante la produzione.

La Time Window 7 è la settima lavorazione effettuata nell'arco temporale di due ore, ha una durata di 8 minuti e 49 secondi. La prova è stata svolta in condizioni di

lavoro normali senza ritardi nelle stazioni manuali, un pezzo è stato scartato durante la produzione.

La Time Window 8 è l'ottava lavorazione eseguita durante il periodo temporale di due ore, ha una durata di 8 minuti e 53 secondi. La prova è stata svolta in un ambiente ideale senza ritardi o pezzi scartati.

La Time Window 9 è la nona e ultima lavorazione della linea temporale osservata di due ore, ha una durata di 8 minuti e 55 secondi. La prova è stata svolta in condizioni di lavoro normali senza ritardi nelle stazioni manuali, un pezzo è stato scartato durante la produzione.
Infine è inserito l'ultimo tempo di errore macchina (ERR 3) della durata di circa 12 minuti, per un fermo macchina totale nell'arco delle due ore di circa 30 minuti.

7.2. Risultati numerici

Dopo aver descritto la struttura di ogni lavorazione svolta all'interno della timeline raffigurata, sono stati calcolati gli indici di prestazione OEE e ROLE delle singole stazioni, l'indice OEE generale dell'intera CPFactory e, seguendo la configurazione descritta nel capitolo precedente, anche l'indice OTE generale dell'intera CPFactory.

In Figura 7.2 viene raffigurata la tabella composta da tutti gli indici appena citati, elencati nelle colonne, per le rispettive finestre temporali, espresse nelle righe. La prima riga raffigura gli indici di prestazione dell'intera produzione svolta nel turno lavorativo di due ore.

	OTE_CPF[%]	OEE_CPF[%]	OEE_AS/RS[%]	ROLE_PBL[%]	OEE_CAM1[%]	ROLE_RW[%]	OEE_CAM2[%]	OEE_COVER[%]
Total Test (2h)	58	56	66	54	63	69	63	70
Time Window 1	95	91	90	89	88	100	91	93
Time Window 2	91	88	87	86	82	97	83	84
Time Window 3	67	71	90	62	90	96	91	92
Time Window 4	21	24	77	20	62	64	63	88
Time Window 5	66	70	84	89	80	45	85	87
Time Window 6	58	61	78	87	81	32	82	83
Time Window 7	82	80	80	77	73	82	75	87
Time Window 8	93	89	82	88	81	98	85	85
Time Window 9	76	73	81	71	71	78	69	84

Figura 7.2.: Indici OEE e OTE della prova di validazione finale.

Gli indici di prestazione delle varie Time Window sono coerenti con la descrizione delle lavorazioni inserita nella sezione precedente. Infatti osservando la T.W.1, la quale è una lavorazione "ottimale", è possibile notare come tutti gli indici di prestazione siano alquanto elevati (quasi tutti superiori al 90%). Nella T.W.2, pur essendo un'altra lavorazione ottimale, per via di quei ritardi involontari introdotti nella linea di produzione gli indici di prestazione risultano essere leggermente più bassi.

Una situazione più interessante avviene osservando la T.W.3 e T.W.4 in quanto, con una lavorazione sulla stazione Pick By Light pari al doppio e al triplo del normale tempo di produzione, è possibile notare come l'abbassamento di tale indice incida sugli indici OTE_CPF (67% per T.W.3 e 21% per T.W.4) e OEE_CPF (71% per

T.W.3 e 24% per T.W.4). In realtà il valore di ROLE_PBL nella T.W.3 dovrebbe aggirarsi intorno al 50%, ma introdurre ritardi volontari con tempi estremamente accurati non è un'operazione prevedibile con precisione, dunque in realtà nel caso di T.W.3 i tempi di lavorazione sono stati leggermente inferiori al doppio del tempo di produzione normale per la stazione Pick By Light e di conseguenza si ha un valore maggiore del 50% (ovvero pari al 62%). Nella T.W.4 inoltre sono stati scartati due pezzi durante la produzione, è possibile confermare questa affermazione osservando che tutti gli indici di prestazione delle singole stazioni coinvolte nella qualità dei pezzi scartati assumono valori inferiori (ROLE_PBL = 20%, OEE_CAM1 = 62%, ROLE_RW = 64%, OEE_CAM2 = 63%).

Nella T.W.5 e T.W.6 valgono le stesse considerazioni fatte per le due precedenti lavorazioni, avendo aumentato i tempi di lavorazione nella stazione manuale, rispettivamente del doppio e del triplo rispetto al normale tempo di produzione, i valori degli indici di prestazione di ROLE_RW (rispettivamente 45% e 32%) vanno ad abbassare gli indici generali OEE (70% per T.W.5 e 61% per T.W.6) e OTE (66% per T.W.5 e 58% per T.W.6) della CPFactory.

Nelle T.W.7 e T.W.9 si hanno situazioni simili in quanto entrambe le lavorazioni hanno un pezzo di scarto al termine della produzione e quindi si ripresenta la situazione già descritta per la T.W.4. Gli indici in T.W.9 sono ulteriormente inferiori in quanto il tempo di produzione è stato lievemente più lungo e di conseguenza meno efficiente.
La T.W.8 è riconducibile al comportamento della T.W.2, ha le stesse caratteristiche di produzione e di conseguenza anche gli indici di prestazione delle stazioni singole e della intera CPFactory sono pressoché simili.

7.3. Considerazioni sulla prova di validazione finale

In questo capitolo è stata descritta nel dettaglio una prova di un processo produttivo, utilizzando la CPFactory, con tempi di produzione consistenti. Dopo aver esposto nel dettaglio la composizione di questa prova nelle relative finestre temporali sono stati mostrati gli indici di prestazione delle singole stazioni e della macchina complessiva.

I valori di OTE_CPF e OEE_CPF, sia nella prova complessiva che nelle singole finestre temporali, riportano complessivamente la corretta condizione generale della macchina mantenendo una certa coerenza tra di loro. Una osservazione da fare è che l'indice OEE_CPF, lungo l'analisi delle varie prove, appare più attenuato rispetto all'OTE_CPF. Anche nelle situazioni in cui gli indici delle singole stazioni si abbassano di un valore considerevole rimane sempre più elevato rispetto all'OTE_CPF. Questo probabilmente è dovuto alla natura del calcolo OEE generale in quanto considera per il valore di Efficienza il tasso medio di elaborazione teorico e reale del ciclo di produzione. L'indice OTE_CPF invece segue di più l'andamento degli indici di prestazione delle singole stazioni, poiché le formule per l'OTE utilizzano proprio questi ultimi, di conseguenza se si presentano indici bassi lungo la produzione il

valore finale di OTE_CPF viene influenzato maggiormente descrivendo meglio la condizione generale della macchina.

Capitolo 8.

Interfaccia grafica su pagina web dinamica

Come già accennato precedentemente, è stata sviluppata anche una interfaccia grafica dove è possibile visualizzare i dati elaborati dallo script in Python e raffigurati attraverso dei grafici direttamente dal computer in cui è presente il database locale *"MyDatabase"*.

La scelta di creare una GUI (Graphical User Interface) è nata dalla necessità di mostrare all'utente le prestazioni della CPFactory in una versione che risultasse più semplice da comprendere piuttosto che osservare tanti valori inseriti in una tabella.

Inoltre è stato deciso di visualizzarla via web, dunque un linguaggio di scripting ampiamente utilizzato (anche per questo elaborato) per lo sviluppo web è PHP, questo linguaggio poi offre un'ampia compatibilità con database popolari come MySQL, questo consente di creare applicazioni web dinamiche che possono interagire con i dati memorizzati nei database.
Per fare ciò è stata sfruttata la piattaforma *XAMPP*, è un pacchetto software che include un ambiente di sviluppo web completo per creare e testare applicazioni web localmente. E' ampiamente utilizzato per configurare un server locale sul proprio computer e sviluppare applicazioni PHP e MySQL senza dover caricarle su un server remoto. Quindi mediante XAMPP è possibile avere un application server capace di interpretare pagine web dinamiche.

Dunque, dopo aver installato XAMPP sul computer associato alla CPFactory, per la visualizzazione della pagina web che raffigura l'interfaccia grafica è necessario avviare i server di *Apache* e *MySQL* presenti all'avvio dell'applicazione, successivamente bisogna aprire il browser e nella barra di ricerca digitare:

- `"localhost:8080/phpmyadmin"` per visualizzare il database locale *MyDatabase* con la relativa tabella *tblValueNew* e i dati al suo interno.
- `"localhost:8080/grafico.php"` per visualizzare l'interfaccia grafica (Figura 8.1) sviluppata apposta per la visualizzazione dei valori prelevati dalla tabella *tblValueNew* di *MyDatabase*.

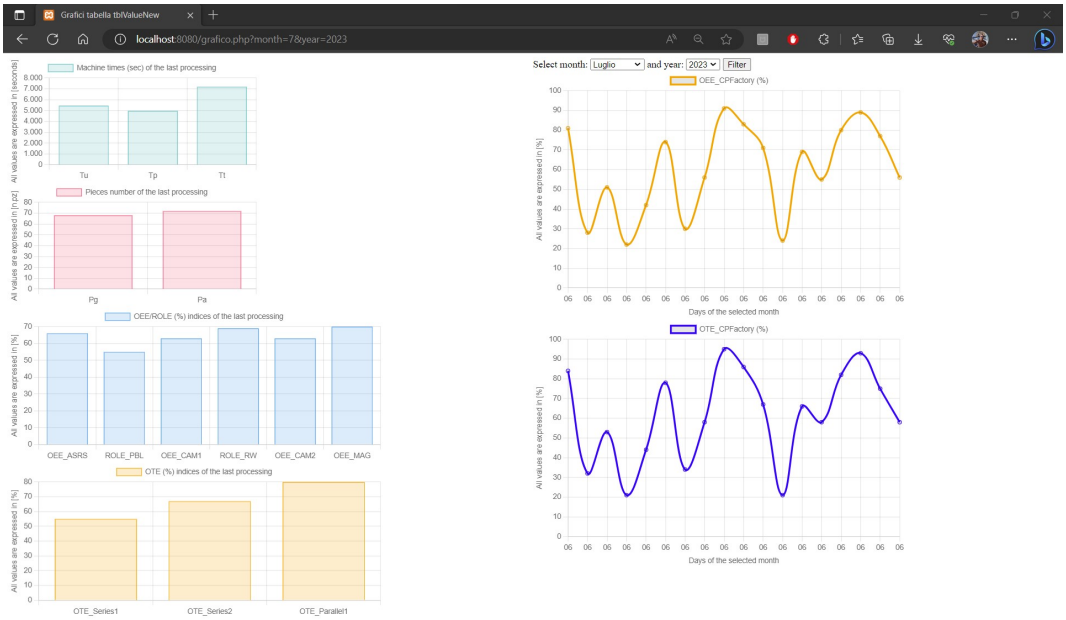


Figura 8.1.: Interfaccia grafica.

In Figura 8.1 è mostrata la visualizzazione dei dati registrati nel database locale *MyDatabase*. I grafici a barre posti sulla sinistra della schermata raffigurano le statistiche dell'ultima misurazione effettuata (Figura 8.2), dall'alto verso il basso: tempi di lavorazione della macchina, pezzi lavorati dalla macchina, indici di prestazione OEE e ROLE delle singole stazioni, indici OTE dei sottosistemi utilizzati.

Figura 8.2.: Interfaccia grafica - parte sinistra.

Mentre i grafici a linea posizionati a destra della finestra (Figura 8.3), dopo aver impostato mese e anno come filtro, raffigurano uno storico nel mese-anno selezionato delle misurazioni effettuate dell'indice OEE e dell'indice OTE dell'intera CPFactory; sull'asse delle ascisse vengono espressi i giorni in cui il dato è stato registrato nel database (dunque nel caso dovessero esserci più valori con lo stesso giorno è perché quelle misurazioni sono state registrate quello stesso giorno).

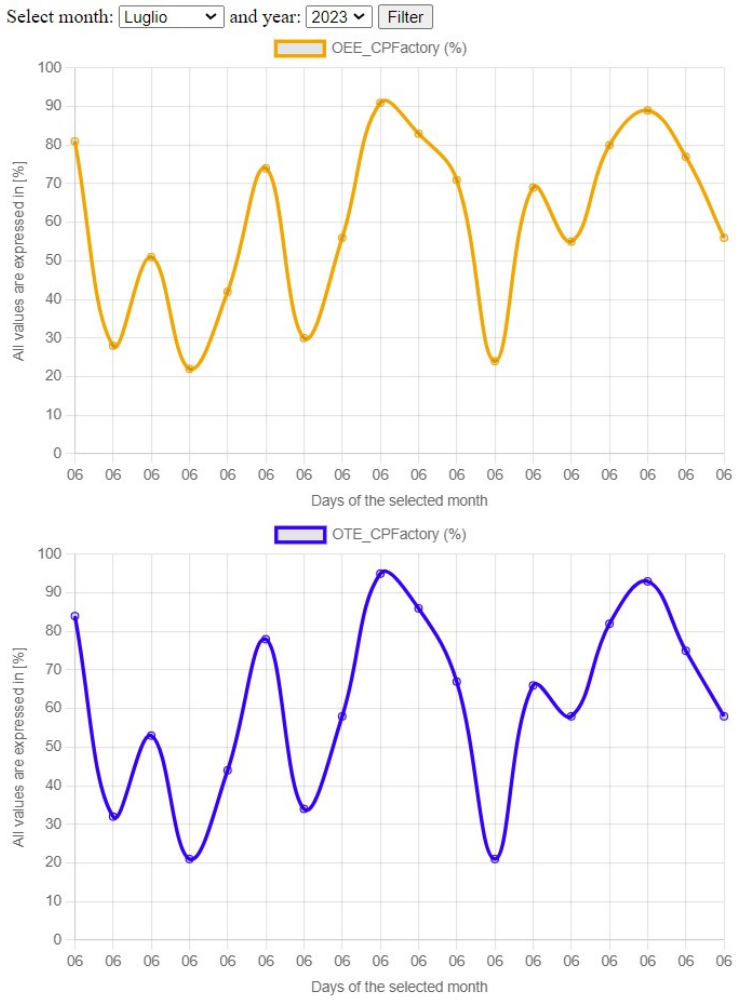


Figura 8.3.: Interfaccia grafica - parte destra.

Nota 1: la porta 8080 è impostata di default, se così non dovesse essere si può vedere quale porta è stata assegnata direttamente su XAMPP, o cambiarla.

Nota 2: se l'operazione di installazione dell'applicazione XAMPP sul computer associato alla macchina non riesce ad andare a buon fine, è necessario al primo avvio *creare* il database nella sezione "phpMyAdmin" e nominarlo *"MyDatabase"*, dove poi verrà creata automaticamente dallo script la tabella con i rispettivi dati.

La spiegazione dettagliata del codice si trova in Appendice B.

Capitolo 9.

Conclusioni

Si conclude questo elaborato in cui sono stati ricavati e analizzati gli indicatori chiave di prestazione a livello macchina, a livello lavoratore e a livello di fabbrica da una linea di produzione automatizzata comprendente anche operazioni effettuate da operatori umani.
Sono state fornite tutte le nozioni necessarie alla comprensione dei processi affrontati nell'elaborato, a partire dalla struttura fisica CPFactory, passando al software MES4 utilizzato per la creazione degli ordini e la definizione del work plan per un determinato prodotto, fino alla teoria associata agli indicatori chiave di prestazione utilizzati.
E' stata inserita una breve descrizione per ogni tabella presente nel database della CPFactory, oltre alla mappatura dei parametri presenti nel database associati ai valori degli indicatori di prestazione.
Successivamente è stata verificata la correttezza dei calcoli eseguiti dallo script in Python, finalizzato al calcolo automatico degli indicatori di prestazione acquisendo dati direttamente dal MES4, attraverso un confronto diretto con metodi di calcolo degli indicatori chiave di prestazione più tradizionali.
E' stata fornita una descrizione dettagliata del software sviluppato in Python, usato per calcolare gli indicatori chiave di prestazione dalla CPFactory, attraverso l'uso di diagrammi di flusso e pseudo-codice. Per vedere il codice completo utilizzato per lo sviluppo dell'elaborato è sufficiente leggere l'Appendice A.
In seguito a diversi tentativi di analisi delle modalità di interconnessione è stata dedotta una interconnessione adeguata tra i relativi sottosistemi osservati sulla CPFactory e sono state definite delle linee guida generali per la realizzazione di interconnessioni utilizzando le quattro configurazioni principali per ricavare l'indicatore di prestazione a livello di fabbrica OTE.
Infine è stata eseguita una prova finale composta da un tempo di produzione elevato in cui sono stati messi a confronto gli indici OEE e OTE generali della macchina, confermando che i due indici rispecchiano correttamente le prestazioni della linea di produzione automatizzata.
Si è voluto creare poi una interfaccia grafica per rendere maggiormente interpretabili i valori ricavati dal software, per vedere il codice utilizzato per la realizzazione è possibile consultare l'Appendice B.

Appendice A.

Codice Python

In questa appendice viene esposto tutto il codice Python usato per la determinazione degli indici di prestazione. Per una maggiore comprensione vedere prima i diagrammi di flusso e lo pseudo-codice al Capitolo 5.

```
1  import csv
2  import time
3  from datetime import datetime
4  from importlib import reload
5  import pandas as pd
6  import pyodbc
7  import mysql.connector
8  import date_new
9
10 # Scrittura dei dati ordinati su un file csv
11 def tabCsv(titolo,results):
12     with open(titolo, "w") as f:
13         writer = csv.writer(f, delimiter=",", lineterminator="\
               n")
14         writer.writerows(results)
15
16 def call_to_database():
17     # Il carattere "r" prima del percorso del file indica a
          Python di interpretare il percorso come stringa "grezza"
           senza elaborare i caratteri di escape.
18     conn_str = (r'DRIVER={Microsoft␣Access␣Driver␣(*.mdb,␣*.
          accdb)};'
19                 r'DBQ=C:\MES4\FestoMES.accdb;')
20     myConnection = pyodbc.connect(conn_str)
21     # Esecuzione della query e ordinamento dei dati in base
          alla data
22     myCursor = myConnection.cursor()
23     # Ordinate per data dal piu recente al meno recente
24     query1 = "SELECT␣ONo,OPos,PlanedStart,PlanedEnd,Start,End,
          Error␣FROM␣tblFinOrderPos␣ORDER␣BY␣PlanedStart␣DESC"
25     myCursor.execute(query1)
26     # Recupero dei risultati della query come lista di tuple
27     results1 = myCursor.fetchall()
```

```
28      # Creazione della lista dei risultati ordinati per data
29      results_sorted1 = sorted(results1, key=lambda x: x[2],
            reverse=True)
30      query2 = "SELECT␣TimeStamp,AutomaticMode,ManualMode,Busy,
            Reset,ErrorL0,ErrorL1,ErrorL2␣FROM␣tblMachineReport␣
            ORDER␣BY␣TimeStamp␣DESC"
31      myCursor.execute(query2)
32      results2 = myCursor.fetchall()
33      results_sorted2 = sorted(results2, key=lambda x: x[0],
            reverse=True)
34      query3 = "SELECT␣PlanedStart,PlanedEnd,Start,End,ResourceID
            ,ErrorStep,ErrorRetVal,TransportTime␣FROM␣tblFinStep␣
            ORDER␣BY␣PlanedStart␣DESC"
35      myCursor.execute(query3)
36      results3 = myCursor.fetchall()
37      results_sorted3 = sorted(results3, key=lambda x: x[0],
            reverse=True)
38      # Salva i dati ordinati sul nuovo file
39      tabCsv("tblMachineReport.csv", results_sorted2)
40      tabCsv("tblFinOrderPos.csv", results_sorted1)
41      tabCsv("tblFinStep.csv", results_sorted3)
42      # Chiusura della connessione
43      myConnection.close()
44
45  def call_to_MyDatabase(tempo_Tu,tempo_Tp,tempo_Tt,avgPlaned,
        avgReal,pezziGiusti,pezziTotali,disp,eff,qual,indice_OEE,
        oee1,role_pbl,oee3,role_rw,oee5,oee6,ote_serie1,ote_serie2,
        ote_esp1,ote_ass1,ote_tot):
46      myNewConnection = mysql.connector.connect(
47          host="localhost",
48          user="root",
49          database="MyDatabase"
50      )
51      nome_tabella = "tblValueNew"
52      myNewCursor = myNewConnection.cursor()
53      # Creazione della tabella se non esiste gia
54      myNewCursor.execute(f"CREATE␣TABLE␣IF␣NOT␣EXISTS␣{
            nome_tabella}␣("
55          f"thisTimeStamp␣DATETIME," # Tempo di registrazione
                valore
56          f"Tu␣DOUBLE,"              # Tempo di uptime
57          f"Tp␣DOUBLE,"              # Tempo di produzione
58          f"Tt␣DOUBLE,"              # Tempo totale di
                osservazione
59          f"RavgTH␣DOUBLE,"          # Tasso medio di
                elaborazione teorico
```

```
60          f"RavgA␣DOUBLE,"            # Tasso medio di
                elaborazione reale
61          f"Pg␣DOUBLE,"               # Pezzi buoni prodotti
62          f"Pa␣DOUBLE,"               # Pezzi totali prodotti
63          f"D␣DOUBLE,"                # Disponibilita OEE generale
64          f"E␣DOUBLE,"                # Efficienza OEE generale
65          f"Q␣DOUBLE,"                # Qualita OEE generale
66          f"OEE_CPF␣DOUBLE,"          # OEE generale
67          f"OEE_ASRS␣DOUBLE,"         # OEE stazione AS/RS
68          f"ROLE_PBL␣DOUBLE,"         # ROLE stazione PBL
69          f"OEE_CAM1␣DOUBLE,"         # OEE stazione Camera 1
70          f"ROLE_RW␣DOUBLE,"          # ROLE stazione Rework
71          f"OEE_CAM2␣DOUBLE,"         # OEE stazione Camera 2
72          f"OEE_MAG␣DOUBLE,"          # OEE stazione Magazzino
73          f"OTE_Series1␣DOUBLE,"      # OTE serie 1
74          f"OTE_Series2␣DOUBLE,"      # OTE serie 2
75          f"OTE_Parallel1␣DOUBLE,"    # OTE parallelo
76          f"OTE_CPF␣DOUBLE)")         # OTE totale della CPF
77      myNewCursor.execute(
78          'INSERT␣INTO␣tblValueNew␣(thisTimeStamp,Tu,Tp,Tt,RavgTH
                ,RavgA,Pg,Pa,D,E,Q,OEE_CPF,OEE_ASRS,ROLE_PBL,
                OEE_CAM1,ROLE_RW,OEE_CAM2,OEE_MAG,OTE_Series1,
                OTE_Series2,OTE_Parallel11,OTE_CPF)␣VALUES␣(%s,%s,%s
                ,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s
                ,%s,%s)',(datetime.now(),tempo_Tu,tempo_Tp,tempo_Tt,
                round(avgPlaned,2),round(avgReal,2),pezziGiusti,
                pezziTotali,round(disp),round(eff),round(qual),round
                (indice_OEE),round(oee1),round(role_pbl),round(oee3)
                ,round(role_rw),round(oee5),round(oee6),round(
                ote_serie1),round(ote_serie2),round(ote_parallel),
                round(ote_tot)))
79      # Commit dei cambiamenti
80      myNewConnection.commit()
81      # Chiusura della connessione
82      myNewConnection.close()
83
84  def load_settings(path_file):
85      # Apre il file di settings in modalita lettura
86      with open(path_file, "r") as file:
87          contenuto = file.read()
88      # Suddivide il contenuto del file di testo riga per riga
89      valori = contenuto.split("\n")
90      # Suddivide ogni riga per il carattere '=' e prende il
            valore dopo di esso, eliminando spazi bianchi con 'strip
            ' e convertendo il valore in intero
91      turno_di_lavoro = int(valori[0].split("=")[1].strip())
92      perdite_strutturali1 = int(valori[1].split("=")[1].strip())
```

```
93      perdite_anomale1 = int(valori[2].split("=")[1].strip())
94      perdite_strutturali2 = int(valori[3].split("=")[1].strip())
95      perdite_anomale2 = int(valori[4].split("=")[1].strip())
96      # 1 = PBL, 2 = RW
97      return turno_di_lavoro, perdite_strutturali1,
            perdite_anomale1, perdite_strutturali2, perdite_anomale2
98
99  def calcolo_OEE(tempo_Tu,tempo_Tp,tempo_Tt,avgPlaned,avgReal,
        pezziGiusti,pezziTotali):
100     # Calcolo dell'indice OEE con la formula data
101     OEE = (tempo_Tu/tempo_Tt)*(tempo_Tp/tempo_Tu)*(avgReal/
            avgPlaned)*(pezziGiusti/pezziTotali)*100
102     # Disponibilita
103     A = tempo_Tu/tempo_Tt*100
104     # Efficienza
105     E = (tempo_Tp/tempo_Tu)*(avgReal/avgPlaned)*100
106     # Qualita
107     Q = pezziGiusti/pezziTotali*100
108     return OEE,A,E,Q
109
110 # OT = Tempo Operativo, MWL = Perdite dovute alla macchina o al
        posto di lavoro
111 def calcolo_OWE(ot,mwl):
112     owe = (ot-mwl)/ot       # OWE = Inefficienze del lavoratore
113     return owe
114
115 # EAT = Tempo Disponibile Effettivo, MDL = Perdite causate
        dalla gestione
116 def calcolo_ME(eat,mdl):
117     me = (eat-mdl)/eat      # ME = Inefficienze nella gestione
118     return me
119
120 # NAT = Tempo Netto Disponibile, ANL = Perdite Anomale
121 def calcolo_WA(nat,anl):
122     wa = (nat-anl)/nat      # WA = Disponibilita del lavoratore
123     return wa
124
125 def calcolo_ROLE(owe,me,wa):
126     role = (owe*me*wa)*100
127     return role
128
129 # 'lista' dei parametri ROLE e 'lista' dei tassi di
        elaborazione teorici
130 def calcolo_OTE_serie(ROLE_list, R_th_list):
131     # Calcola il valore minimo tra ROLE[i] * R_th[i] per ogni
            indice 'i'
```

```
132     min_ROLE_R_th = min([ROLE_list[i] * R_th_list[i] for i in
            range(len(ROLE_list))])
133     min_R_th = min(R_th_list)
134     OTE_serie = (min_ROLE_R_th / min_R_th)
135     return OTE_serie
136 
137 def calcolo_OTE_parallelo(ROLE_list, R_th_list):
138     sum_ROLE_R_th = sum([ROLE_list[i] * R_th_list[i] for i in
            range(len(ROLE_list))])
139     sum_R_th = sum(R_th_list)
140     OTE_parallelo = (sum_ROLE_R_th / sum_R_th)
141     return OTE_parallelo
142 
143 def calcolo_OTE_assemblaggio(ROLE_list, R_th_list, K_list,
        ROLE_a, R_a_th):
144     min_NUM1 = min([ROLE_list[i] * (R_th_list[i] / K_list[i]) *
             ROLE_a for i in range(len(ROLE_list))])
145     min_NUM = min(min_NUM1, ROLE_a * R_a_th)
146     min_DEN1 = min([R_th_list[i] / K_list[i] for i in range(len
            (ROLE_list))])
147     min_DEN = min(min_DEN1, R_a_th)
148     OTE_assemblaggio = min_NUM / min_DEN
149     return OTE_assemblaggio
150 
151 def calcolo_OTE_espansione(ROLE_list, R_th_list, K_list, ROLE_e
        , R_e_th):
152     sum_NUM = sum([min(ROLE_e * R_e_th * K_list[i] * ROLE_list[
            i], ROLE_list[i] * R_th_list[i]) for i in range(len(
            ROLE_list))])
153     sum_DEN = sum([min(R_e_th * K_list[i], R_th_list[i]) for i
            in range(len(ROLE_list))])
154     OTE_espansione = sum_NUM / sum_DEN
155     return OTE_espansione
156 
157 def main():
158     while 1:
159         # Chiamata al database della Festo
160         call_to_database()
161         path_MyFile = r"C:\CPF_Project\input.txt"
162         # Lettura dal file di input dei dati principali
163         turno, perdite_SL_PBL, perdite_ANL_PBL, perdite_SL_RW,
                perdite_ANL_RW =
164         load_settings(path_MyFile)
165         # Date di inizio e fine osservazione prese dal modulo "
                date_new"
166         start_date, end_date = date_new.caricamento_date(turno)
167 
```

```
168         try:
169             # Leggi il file CSV di FinOrderPos senza la prima
                riga d'intestazione, uso un dataframe
170             df_FOP = pd.read_csv('tblFinOrderPos.csv', header=
                None)
171             # Specifica dei nomi delle colonne manualmente
172             df_FOP.columns = ['ONo','OPos','PlanedStart','
                PlanedEnd','Start','End','Error']
173             # Converto le date da formato stringa in oggetti
                datetime
174             df_FOP['PlanedStart'] = pd.to_datetime(df_FOP['
                PlanedStart'])
175             df_FOP['PlanedEnd'] = pd.to_datetime(df_FOP['
                PlanedEnd'])
176             df_FOP['Start'] = pd.to_datetime(df_FOP['Start'])
177             df_FOP['End'] = pd.to_datetime(df_FOP['End'])
178             # Prendo un range di tempo dal tempo attuale a 30
                minuti prima
179             df_filteredFOP = df_FOP.loc[(df_FOP['PlanedStart']
                > start_date) & (df_FOP['PlanedStart'] <
                end_date)]
180             pezziGiusti = 0     # Pg
181             pezziTotali = 0     # Pa
182             # Calcolo i pezzi totali e i pezzi giusti in base
                al periodo di tempo scelto
183             for row in df_filteredFOP['Error']:
184                 # Se l'indice assume valore 'false' allora il
                    pezzo va bene
185                 if not row:
186                     pezziGiusti+=1
187                 pezziTotali+=1
188             # Per conoscere il numero di elementi nel dataframe
189             countRowsFOP = df_filteredFOP['ONo'].count()
190             # Elenco di ogni periodo di tempo pianificato,
                creazione di un nuovo dataframe
191             df_TimePlanedFOP = df_filteredFOP['PlanedEnd'] -
                df_filteredFOP['PlanedStart']
192             # Elenco di ogni periodo di tempo reale trascorso,
                sono ancora dei dataframe
193             df_TimeRealFOP = df_filteredFOP['End'] -
                df_filteredFOP['Start']
194             # Ravg(th), ricavato sommando il dataframe adeguato
                 e dividendo per il numero di righe del
                dataframe
195             avgPlaned = (df_TimePlanedFOP.sum()/countRowsFOP).
                total_seconds()
196             # Ravg(a)
```

```
197             avgReal = (df_TimeRealFOP.sum()/countRowsFOP).
                    total_seconds()
198
199             # Questa tabella mi serve per calcolare le perdite
                    nella stazione pick by light e rework. (work
                    time = planedEnd - planedStart)
200             df_FS = pd.read_csv('tblFinStep.csv', header=None)
201             df_FS.columns = ['PlanedStart','PlanedEnd','Start',
                    'End','ResourceID','ErrorStep','ErrorRetVal','
                    TransportTime']
202             df_FS['PlanedStart'] = pd.to_datetime(df_FS['
                    PlanedStart'])
203             df_FS['PlanedEnd'] = pd.to_datetime(df_FS['
                    PlanedEnd'])
204             df_FS['Start'] = pd.to_datetime(df_FS['Start'])
205             df_FS['End'] = pd.to_datetime(df_FS['End'])
206             df_filteredFS = df_FS.loc[(df_FS['PlanedStart'] >
                    start_date) & (df_FS['PlanedStart'] < end_date)]
207             # Modulo applicativo 1 (AS/RS):
208             avgR1_planed_temp = 0
209             avgR1_real_temp = 0
210             # Modulo applicativo 2 (Pick By Light):
211             wpTimePlanedPBL_temp = 0
212             wpTimeRealPBL_temp = 0
213             # Modulo applicativo 3 (camera 1):
214             avgR3_planed_temp = 0
215             avgR3_real_temp = 0
216             # Modulo applicativo 4 (Rework):
217             wpTimePlanedRW_temp = 0
218             wpTimeRealRW_temp = 0
219             # Modulo applicativo 5 (camera 2):
220             avgR5_planed_temp = 0
221             avgR5_real_temp = 0
222             # Modulo applicativo 6 (magazzino gusci anteriori):
223             avgR6_planed_temp = 0
224             avgR6_real_temp = 0
225             # Contatori utilizzati:
226             countForPBL = 0
227             countForRW = 0
228             countForMag = 0
229             # Variabile aggiuntiva per risolvere un problema
                    con la camera2:
230             transportTime5 = 0
231             for index, row in df_filteredFS.iterrows():
232                 # Per ROLE del Pick By Light
233                 if row['ResourceID'] == 2:
```

```
234                     wpTimePlanedPBL_temp += (row['PlanedEnd'] -
                            row['PlanedStart']).total_seconds() +
                            row['TransportTime']
235                     wpTimeRealPBL_temp += (row['End'] - row['
                            Start']).total_seconds() + row['
                            TransportTime']
236                 # Esclude le righe vuote nelle date di Start ed
                        End, altrimenti sbaglia la media (per ROLE
                        del Rework)
237                 if row['ResourceID'] == 4 and not pd.isnull(row
                        ['Start']):
238                     wpTimePlanedRW_temp += (row['PlanedEnd'] -
                            row['PlanedStart']).total_seconds() +
                            row['TransportTime']
239                     wpTimeRealRW_temp += (row['End'] - row['
                            Start']).total_seconds() + row['
                            TransportTime']
240                 # Incrementa il contatore solo quando la camera
                        1 fallisce l'ispezione del pezzo
241                 if row['ResourceID'] == 3 and row['ErrorRetVal'
                        ] == 5050:
242                     countForPBL += 1
243                 # Incrementa il contatore solo quando la camera
                        2 fallisce l'ispezione del pezzo
244                 if row['ResourceID'] == 5 and row['ErrorRetVal'
                        ] == 5050:
245                     countForRW += 1
246                 # Serve per associare un errore nella 'qualita'
                        al modulo magazzino gusci anteriori
247                 if row['ResourceID'] == 4 and row['ErrorRetVal'
                        ] == 5000:
248                     countForMag += 1
249                 # Per il calcolo dell'OEE dell modulo AS/RS
250                 if row['ResourceID'] == 1 and not pd.isnull(row
                        ['Start']):
251                     avgR1_planed_temp += ((row['PlanedEnd'] -
                            row['PlanedStart']).total_seconds()) +
                            row['TransportTime']
252                     avgR1_real_temp += ((row['End'] - row['
                            Start']).total_seconds()) + row['
                            TransportTime']
253                 # Per il calcolo dell'OEE del modulo con la
                        camera 1
254                 if row['ResourceID'] == 3 and not pd.isnull(row
                        ['Start']):
255                     avgR3_planed_temp += ((row['PlanedEnd'] -
                            row['PlanedStart']).total_seconds()) +
```

```
                        row['TransportTime']
256                     avgR3_real_temp += ((row['End'] - row['
                        Start']).total_seconds()) + row['
                        TransportTime']
257                 # Per il calcolo dell'OEE del modulo con la
                        camera 2
258                 if row['ResourceID'] == 5 and not pd.isnull(row
                        ['Start']):
259                     avgR5_planed_temp += ((row['PlanedEnd'] -
                        row['PlanedStart']).total_seconds()) +
                        row['TransportTime']
260                     avgR5_real_temp += ((row['End'] - row['
                        Start']).total_seconds()) + row['
                        TransportTime']
261                 if row['ResourceID'] == 5:
262                     transportTime5 = row['TransportTime']
263                 # Per il calcolo dell'OEE del modulo Magazzino
                        gusci anteriori
264                 if row['ResourceID'] == 6 and not pd.isnull(row
                        ['Start']):
265                     avgR6_planed_temp += ((row['PlanedEnd'] -
                        row['PlanedStart']).total_seconds()) +
                        row['TransportTime']
266                     avgR6_real_temp += ((row['End'] - row['
                        Start']).total_seconds()) + row['
                        TransportTime']
267             # Media dei tempi pianificati di lavorazione della
                    stazione Pick By Light
268             wpTimePlanedPBL = wpTimePlanedPBL_temp/pezziTotali
269             # Media dei tempi reali di lavorazione della
                    stazione Pick By Light
270             wpTimeRealPBL = wpTimeRealPBL_temp/pezziTotali
271             # Media dei tempi pianificati di lavorazione della
                    stazione Rework, si usano i pezzi giusti perche
                    le operazioni di rework sugli scarti non sono
                    contemplate nel database
272             wpTimePlanedRW = wpTimePlanedRW_temp/pezziGiusti
273             # Media dei tempi reali di lavorazione della
                    stazione di rilavorazione rework
274             wpTimeRealRW = wpTimeRealRW_temp/pezziGiusti
275             # Prende il valore 'wpTimeRealPBL' e lo moltiplica
                    per le volte che ho un pezzo scartato nella
                    camera 1, ottengo le perdite MWL per il ROLE1.
276             perdite_MWL_PBL = wpTimeRealPBL * countForPBL
277             perdite_MWL_RW = wpTimeRealRW * countForRW
278             # In questo caso prendo la variabile 'temp' perche
                    questa perdita non riguarda la singola
```

```
            lavorazione ma la gestione delle lavorazioni
            complessive.
279         perdite_MDL_PBL = wpTimeRealPBL_temp -
            wpTimePlanedPBL_temp
280         perdite_MDL_RW = wpTimeRealRW_temp -
            wpTimePlanedRW_temp
281         # Tassi medi di elaborazione teorici e reali delle
            stazioni della CPF:
282         # Puo cambiare in base a quanti pezzi errati ci
            sono, se pezzo errato cambia percorso e cambiano
             i tempi
283         avgR1_planed = avgR1_planed_temp / pezziTotali
284         avgR1_real = avgR1_real_temp / pezziTotali
285         avgR3_planed = avgR3_planed_temp / pezziTotali
286         avgR3_real = avgR3_real_temp / pezziTotali
287         # Puo essere che se non ci sono pezzi errati il
            valore sia nullo, in tal caso:
288         if countForPBL != 0:
289             # La camera 2 funziona solo quando i pezzi sono
                 errati nel PBL, dunque tutte le volte che
                la camera 1 ha dato errore
290             avgR5_planed = avgR5_planed_temp / countForPBL
291             avgR5_real = avgR5_real_temp / countForPBL
292         else:
293             # Imposto solo il tempo di trasporto, anche se
                non viene utilizzata.
294             avgR5_planed = transportTime5
295             avgR5_real = transportTime5
296         # Il Magazzino dei gusci anteriori funziona solo
            quando i pezzi sono corretti, altrimenti non
            posiziona nulla
297         avgR6_planed = avgR6_planed_temp / pezziGiusti
298         avgR6_real = avgR6_real_temp / pezziGiusti
299         pg_1 = pezziTotali    # Pezzi giusti del modulo 1
300         pa_1 = pezziTotali    # Pezzi totali del modulo 1
301         # Pezzi giusti del modulo 3
302         pg_3 = pezziTotali-countForPBL
303         pa_3 = pezziTotali    # Pezzi totali del modulo 3
304         # Pezzi giusti del modulo 5
305         pg_5 = pezziTotali-countForRW
306         pa_5 = pezziTotali    # Pezzi totali del modulo 5
307         # Pezzi giusti del modulo 6
308         pg_6 = pezziTotali-countForMag
309         pa_6 = pezziTotali    # Pezzi totali del modulo 6
310
311         # Leggo il file csv di MachineReport per trovare i
            parametri rimanenti
```

```
312             df_MR = pd.read_csv('tblMachineReport.csv', header=
                    None)
313             df_MR.columns = ['TimeStamp','AutomaticMode','
                    ManualMode','Busy','Reset','ErrorL0','ErrorL1','
                    ErrorL2']
314             df_MR['TimeStamp'] = pd.to_datetime(df_MR['
                    TimeStamp'])
315             # Creo un dataframe filtrato nel periodo
                    selezionato all'inizio
316             df_filteredMR = df_MR.loc[(df_MR['TimeStamp'] >
                    start_date) & (df_MR['TimeStamp'] < end_date)]
317             # Resetta la numerazione degli indici del dataframe
                    , utile per dopo
318             df_filteredMR = df_filteredMR.reset_index(drop=True
                    )
319             # Tempo di produzione della macchina, sarebbe
                    quando sta in modalita automatica
320             tempo_Tp = 0
321             tempo_Terr = 0  # Tempo di fermo della macchina
322             # Index viene usato per le righe, row per le
                    colonne del dataframe
323             for index, row in df_filteredMR.iterrows():
324                 # Solo quando sta in automatic
325                 if row['AutomaticMode']:
326                     # Assegna a t_pre il valore della data a
                            quella riga, timestamp() prende il
                            valore in secondi dal 1970
327                     t_down1 = df_filteredMR.loc[index, '
                            TimeStamp'].timestamp()
328                     # Dal momento che ho riassegnato gli indici
                             a partire da 0, se ho -1 significa che
                            sono out of index
329                     if index-1 == -1 or df_filteredMR.loc[index
                            -1, 'AutomaticMode'] == False:
330                         t_upper1 = t_down1
331                     else:
332                         # Assegna a t_post il valore della data
                                alla riga precedente, se sopra
                                finisce il dataframe ritorna errore
                                (out of index)
333                         t_upper1 = df_filteredMR.loc[index-1, '
                                TimeStamp'].timestamp()
334                     # Valore in secondi della differenza tra la
                             coppia
335                     temporary_Tp = (t_upper1 - t_down1)
336                     # Sommo a Tp tutti i tempi in sola modalita
                             automatica
```

```
337                     tempo_Tp += temporary_Tp
338                 if row['ManualMode'] or row['Reset'] or row['
                        ErrorL0'] or row['ErrorL1'] or row['ErrorL2'
                        ]:
339                     t_down2 = df_filteredMR.loc[index, '
                            TimeStamp'].timestamp()
340                     if index-1 == -1:
341                         t_upper2 = t_down2
342                     else:
343                         t_upper2 = df_filteredMR.loc[index-1, '
                                TimeStamp'].timestamp()
344                     temporary_Terr = (t_upper2 - t_down2)
345                     tempo_Terr += temporary_Terr
346             # Tempo totale di osservazione della macchina,
                    prende la data di inizio periodo di osservazione
                     e la data di fine dataframe filtrato (ovvero
                    fine osservazione)
347             tempo_Tt = (end_date - start_date).total_seconds()
348             # Tempo di attivita della macchina (uptime)
349             tempo_Tu = tempo_Tt - tempo_Terr
350
351             # Calcolo dell'indice OEE
352             indice_OEE, disp, eff, qual = calcolo_OEE(tempo_Tu,
                    tempo_Tp,tempo_Tt,avgPlaned,avgReal,pezziGiusti,
                    pezziTotali)
353             # Nelle perdite anomale sono state aggiunte anche
                    le perdite dovute a fermi della macchina
354             role_pbl = calcolo_ROLE(calcolo_OWE(
                    wpTimePlanedPBL_temp,perdite_MWL_PBL),
                    calcolo_ME(wpTimeRealPBL_temp,perdite_MDL_PBL),
                    calcolo_WA(tempo_Tt,(perdite_ANL_PBL+tempo_Terr)
                    ))
355             role_rw = calcolo_ROLE(calcolo_OWE(
                    wpTimePlanedRW_temp,perdite_MWL_RW), calcolo_ME(
                    wpTimeRealRW_temp,perdite_MDL_RW), calcolo_WA(
                    tempo_Tt,(perdite_ANL_RW+tempo_Terr)))
356             # Calcolo OEE delle stazioni 1,3,5,6:
357             oee1, disp1, eff1, qual1 = calcolo_OEE(tempo_Tu,
                    tempo_Tp,tempo_Tt,avgR1_planed,avgR1_real,pg_1,
                    pa_1)
358             oee3, disp3, eff3, qual3 = calcolo_OEE(tempo_Tu,
                    tempo_Tp,tempo_Tt,avgR3_planed,avgR3_real,pg_3,
                    pa_3)
359             oee5, disp5, eff5, qual5 = calcolo_OEE(tempo_Tu,
                    tempo_Tp,tempo_Tt,avgR5_planed,avgR5_real,pg_5,
                    pa_5)
```

```
360             oee6, disp6, eff6, qual6 = calcolo_OEE(tempo_Tu,
                    tempo_Tp,tempo_Tt,avgR6_planed,avgR6_real,pg_6,
                    pa_6)
361             # Calcolo OTE:
362             lista_OEE_serie1 = [role_pbl,oee1,oee3]
363             lista_tempi_serie1 = [wpTimePlanedPBL,avgR1_planed,
                    avgR3_planed]
364             OTE_serie1 = calcolo_OTE_serie(lista_OEE_serie1,
                    lista_tempi_serie1)
365             lista_OEE_serie2 = [role_rw,oee5]
366             lista_tempi_serie2 = [wpTimePlanedRW,avgR5_planed]
367             OTE_serie2 = calcolo_OTE_serie(indici_list_serie2,
                    lista_tempi_serie2)
368             lista_OEE_parallelo1 = [100,OTE_serie2]
369             lista_tempi_parallelo1 = [26,(wpTimePlanedRW+
                    avgR5_planed)]
370             OTE_parallelo = calcolo_OTE_parallelo(
                    lista_OEE_parallelo1,lista_tempi_parallelo1)
371             lista_OTE = [OTE_serie1,OTE_parallelo,oee6]
372             lista_tempi_OTE = [sum(lista_tempi_serie1),sum(
                    lista_tempi_parallelo1),avgR6_planed]
373             OTE = calcolo_OTE_serie(lista_OTE1,lista_tempi_OTE1
                    )
374
375             # Chiamata al mio database
376             call_to_MyDatabase(tempo_Tu,tempo_Tp,tempo_Tt,
                    avgPlaned,avgReal,pezziGiusti,pezziTotali,disp,
                    eff,qual,indice_OEE,oee1,role_pbl,oee3,role_rw,
                    oee5,oee6,OTE_serie1,OTE_serie2,OTE_parallelo,
                    OTE)
377             # Pausa del ciclo per 30 minuti (in secondi) diviso
                     3 -> ogni 10 minuti fa la chiamata
378             time.sleep((turno*60)/3)
379             reload(date_new)
380         except ValueError:
381             print('Errore di lettura --> Database vuoto')
382             time.sleep(30)
383             # Usato il modulo "date_new" solo per poter
                    utilizzare la funzione "reload" per ricaricare
                    le nuove date
384             reload(date_new)
385
386 if __name__ == "__main__":
387     main()
```

Appendice B.

Codice PHP

Di seguito viene mostrato il codice PHP, denominato *grafico.php*, sviluppato per la creazione dell'interfaccia grafica. Per la corretta esecuzione dello script, questo file è salvato direttamente all'interno della cartella di XAMPP, ovvero in `"C:\xampp\htdocs\grafico.php"`. Il codice PHP incorpora anche del codice HTML per la struttura della pagina e il rendering dei grafici.

```
1  <?php
2  $host = "localhost";
3  $user = "root";
4  $password = "";
5  $dbname = "MyDatabase";
6  # Oggetto di connessione al database
7  $conn = new mysqli($host, $user, $password, $dbname);
8  if ($conn->connect_error) {
9          die("Connessione al database fallita: " . $conn->
               connect_error);
10 }
11 # Nomi che rappresentano i nomi delle colonne che verranno
      estratte dalla tabella del database
12 $group1Columns = array("Tu", "Tp", "Tt");
13 $group2Columns = array("Pg", "Pa");
14 $group3Columns = array("OEE_ASRS", "ROLE_PBL", "OEE_CAM1", "
      ROLE_RW", "OEE_CAM2", "OEE_MAG");
15 $group4Columns = array("OTE_Serie1", "OTE_Serie2", "
      OTE_Parallel1");
16 # Array multidimensionale inizialmente vuoto, per contenere i
      dati estratti dal database
17 $data = array(
18     'group1' => array(),
19     'group2' => array(),
20     'group3' => array(),
21     'group4' => array()
22 );
23 # Query che prende l'ultima riga per il gruppo dei tempi, "
      implode" unisce i nomi delle colonne in una stringa separata
       da virgole
```

```
24 $sql = "SELECT␣" . implode(",␣", $group1Columns) . "␣FROM␣
       tblValueNew␣ORDER␣BY␣thisTimeStamp␣DESC␣LIMIT␣1";
25 $result = $conn->query($sql);
26 # Verifica se la query ha prodotto almeno una riga di risultati
27 if ($result->num_rows > 0) {
28     # 'fetch_assoc()' utilizzata per estrarre la riga di
          risultati come un array associativo (ogni colonna del
          risultato sara un elemento dell'array e sara accessibile
           utilizzando il nome della colonna come chiave)
29     $row = $result->fetch_assoc();


30     $group1Data = array();
31     # Per ogni colonna, viene assegnato il valore
          corrispondente dalla riga di risultati all'array '
          group1Data'
32     foreach ($group1Columns as $column) {
33         $group1Data[$column] = $row[$column];
34     }
35     #  L'uso delle parentesi quadre [] alla fine dell'
          assegnazione fa si che l'array 'group1Data' venga
          aggiunto come nuovo elemento all'array 'data['group1']'
36     $data['group1'][] = $group1Data;
37 }
38 # Query che prende l'ultima riga per il gruppo dei pezzi
39 $sql = "SELECT␣" . implode(",␣", $group2Columns) . "␣FROM␣
       tblValueNew␣ORDER␣BY␣thisTimeStamp␣DESC␣LIMIT␣1";
40 $result = $conn->query($sql);
41 if ($result->num_rows > 0) {
42     $row = $result->fetch_assoc();
43     $group2Data = array();
44     foreach ($group2Columns as $column) {
45         $group2Data[$column] = $row[$column];
46     }
47     $data['group2'][] = $group2Data;
48 }
49 # Query che prende l'ultima riga per il gruppo degli indici OEE
       e ROLE delle stazioni
50 $sql = "SELECT␣" . implode(",␣", $group3Columns) . "␣FROM␣
       tblValueNew␣ORDER␣BY␣thisTimeStamp␣DESC␣LIMIT␣1";
51 $result = $conn->query($sql);
52 if ($result->num_rows > 0) {
53     $row = $result->fetch_assoc();
54     $group3Data = array();
55     foreach ($group3Columns as $column) {
56         $group3Data[$column] = $row[$column];
57     }
```

```
58      $data['group3'][] = $group3Data;
59  }
60  # Query che prende l'ultima riga per il gruppo degli indici OTE
        della macchina
61  $sql = "SELECT␣" . implode(",␣", $group4Columns) . "␣FROM␣
        tblValueNew␣ORDER␣BY␣thisTimeStamp␣DESC␣LIMIT␣1";
62  $result = $conn->query($sql);
63  if ($result->num_rows > 0) {
64      $row = $result->fetch_assoc();
65      $group4Data = array();
66      foreach ($group4Columns as $column) {
67          $group4Data[$column] = $row[$column];
68      }
69      $data['group4'][] = $group4Data;
70  }
71  # Ottiene il valore del mese dalla query string GET. Se il
        valore e' presente, viene assegnato alla variabile
        $selectedMonth.
72  # In caso contrario, viene utilizzato il valore di default
        restituito dalla funzione date('m'), che restituisce il mese
        corrente.
73  $selectedMonth = isset($_GET['month']) ? $_GET['month'] : date(
        'm');
74  # Ottiene il valore dell'anno dalla query string GET
75  $selectedYear = isset($_GET['year']) ? $_GET['year'] : date('Y'
        );
76  # Query SQL per selezionare i dati di interesse dalla tabella "
        tblValueNew" in base al mese e all'anno selezionati.
77  $query_line = "SELECT␣thisTimeStamp,␣OEE_CPF,␣OTE_CPF␣FROM␣
        tblValueNew␣WHERE␣MONTH(thisTimeStamp)␣=␣$selectedMonth␣AND␣
        YEAR(thisTimeStamp)␣=␣$selectedYear";
78  $result_line = mysqli_query($conn, $query_line);
79  # Verifica se la query ha prodotto dei risultati
80  if (mysqli_num_rows($result_line) > 0) {
81      # Array per i valori delle colonne
82      $days = array();
83      $oeeValues = array();
84      $oteValues = array();
85      # Ciclo sui risultati della query
86      # Ogni riga di risultati viene rappresentata come un array
            associativo nella variabile 'row'
87      while ($row = mysqli_fetch_assoc($result_line)) {
88          # Viene estratto il giorno dalla colonna 'thisTimeStamp
                ', 'strtotime()' viene utilizzata per convertire la
                data nel formato corretto
89          $day = date('d', strtotime($row['thisTimeStamp']));
```

```
90          # Aggiunge il giorno, il valore di OEE e OTE ai
                rispettivi array
91          $days[] = $day;

92          $oeeValues[] = $row['OEE_CPF'];
93          $oteValues[] = $row['OTE_CPF'];
94      }
95  } else {
96      # Se la query non ha prodotto risultati
97      echo "Nessun risultato trovato.";

98  }
99  ?>
100
101 <!DOCTYPE html>
102 <html lang="it">
103 <head>
104         <meta charset="UTF-8">
105         <title>Grafici tabella tblValueNew</title>
106         <meta name="viewport" content="width=device-width, 
                initial-scale=1.0">
107         <script src="https://cdn.jsdelivr.net/npm/chart.js"></
                script>
108         <link rel="stylesheet" type="text/css" href="
                divisioneColonne.css">
109 </head>
110 <body>
111     <div class="chart-container">
112         <div class="chart-column">
113             <canvas id="chart1" width="400" height="200"></
                    canvas>
114             <canvas id="chart2" width="400" height="200"></
                    canvas>
115             <canvas id="chart3" width="600" height="250"></
                    canvas>
116             <canvas id="chart4" width="600" height="250"></
                    canvas>
117         </div>
118         <div class="chart-column">
119             <form action="" method="GET">
120                 <label for="month">Select month:</label>
121                 <select name="month" id="month">
122                     <option value="1" <?php if (isset($_GET['
                            month']) && $_GET['month'] == 1) echo '
                            selected'; ?>>Gennaio</option>
123                     <option value="2" <?php if (isset($_GET['
                            month']) && $_GET['month'] == 2) echo '
```

```
                        selected'; ?>>Febbraio</option>
124                     <option value="3" <?php if (isset($_GET['
                        month']) && $_GET['month'] == 3) echo '
                        selected'; ?>>Marzo</option>
125                     <option value="4" <?php if (isset($_GET['
                        month']) && $_GET['month'] == 4) echo '
                        selected'; ?>>Aprile</option>
126                     <option value="5" <?php if (isset($_GET['
                        month']) && $_GET['month'] == 5) echo '
                        selected'; ?>>Maggio</option>
127                     <option value="6" <?php if (isset($_GET['
                        month']) && $_GET['month'] == 6) echo '
                        selected'; ?>>Giugno</option>
128                     <option value="7" <?php if (isset($_GET['
                        month']) && $_GET['month'] == 7) echo '
                        selected'; ?>>Luglio</option>
129                     <option value="8" <?php if (isset($_GET['
                        month']) && $_GET['month'] == 8) echo '
                        selected'; ?>>Agosto</option>
130                     <option value="9" <?php if (isset($_GET['
                        month']) && $_GET['month'] == 9) echo '
                        selected'; ?>>Settembre</option>
131                     <option value="10" <?php if (isset($_GET['
                        month']) && $_GET['month'] == 10) echo '
                        selected'; ?>>Ottobre</option>
132                     <option value="11" <?php if (isset($_GET['
                        month']) && $_GET['month'] == 11) echo '
                        selected'; ?>>Novembre</option>
133                     <option value="12" <?php if (isset($_GET['
                        month']) && $_GET['month'] == 12) echo '
                        selected'; ?>>Dicembre</option>
134                 </select>
135                 <label for="year">and year:</label>
136                 <select name="year" id="year">
137                     <?php
138                     $selectedYear = isset($_GET['year']) ?
                        $_GET['year'] : date('Y');
139                     $currentYear = date('Y');
140                         # Modifica il valore se vuoi un range
                            di anni diverso
141                     $startYear = $currentYear - 5;
142
143                     for ($year = $startYear; $year <=
                        $currentYear; $year++) {
144                         $selected = $selectedYear == $year ? '
                            selected' : '';
```

```
145                         echo "<option value=\"$year\" $selected
                                >$year</option>";
146                     }
147                     ?>
148                 </select>
149                 <button type="submit">Filtra</button>
150             </form>
151             <?php if (isset($days) && isset($oeeValues) &&
                    isset($oteValues) && isset($_GET['month']) &&
                    isset($_GET['year'])): ?>
152                 <canvas id="lineChart1" width="600" height="400
                    "></canvas>
153                 <canvas id="lineChart2" width="600" height="400
                    "></canvas>
154         </div>
155     </div>
156
157 <script>
158 window.onload = function() {
159     # Converte l'array '$data' in formato JSON
160     var data = <?php echo json_encode($data); ?>;

161     # Vengono estratti i valori dei tre gruppi di dati dall'
        oggetto 'data'
162     var group1Data = Object.values(data.group1[0]);
163     var group2Data = Object.values(data.group2[0]);
164     var group3Data = Object.values(data.group3[0]);
165     var group4Data = Object.values(data.group4[0]);
166     # Vengono estratti i nomi delle colonne dei tre gruppi di
        dati
167     var group1Labels = Object.keys(data.group1[0]);
168     var group2Labels = Object.keys(data.group2[0]);
169     var group3Labels = Object.keys(data.group3[0]);
170     var group4Labels = Object.keys(data.group4[0]);
171     # Vengono ottenuti i contesti dei tre grafici a barre dall'
        HTML
172     var ctx1 = document.getElementById('chart1').getContext('2d
        ');
173     var ctx2 = document.getElementById('chart2').getContext('2d
        ');
174     var ctx3 = document.getElementById('chart3').getContext('2d
        ');
175     var ctx4 = document.getElementById('chart4').getContext('2d
        ');
176     var chart1 = new Chart(ctx1, {
177         type: 'bar',
178         data: {
```

```
179             labels: group1Labels,
180             datasets: [{
181                 label: 'Machine times (sec) of the last
                        processing',
182                 data: group1Data,
183                 backgroundColor: 'rgba(75, 192, 192, 0.2)',
184                 borderColor: 'rgba(75, 192, 192, 1)',
185                 borderWidth: 1
186             }]},
187         options: {
188             responsive: false,
189             maintainAspectRatio: false,
190             scales: {
191                 y: {beginAtZero: true},
192                 title: {
193                         display: true,
194                         text: 'All values are expressed in [
                                seconds]'
195                     }},
196             plugins: {
197                 legend: {
198                     display: true,
199                     position: 'top'
200                 }}}
201     });
202     var chart2 = new Chart(ctx2, {
203         type: 'bar',
204         data: {
205             labels: group2Labels,
206             datasets: [{
207                 label: 'Pieces number of the last processing',
208                 data: group2Data,
209                 backgroundColor: 'rgba(255, 99, 132, 0.2)',
210                 borderColor: 'rgba(255, 99, 132, 1)',
211                 borderWidth: 1
212             }]},
213         options: {
214             responsive: false,
215             maintainAspectRatio: false,
216             scales: {
217                 y: {beginAtZero: true},
218                 title: {
219                         display: true,
220                         text: 'All values are expressed in [n.
                                pz]'
221                     }},
222             plugins: {
```

```
223                 legend: {
224                     display: true,
225                     position: 'top'
226                 }}}
227     });
228     var chart3 = new Chart(ctx3, {
229         type: 'bar',
230         data: {
231             labels: group3Labels,
232             datasets: [{
233                 label: 'OEE/ROLE (%) indices of the last
                        processing',
234                 data: group3Data,
235                 backgroundColor: 'rgba(54, 162, 235, 0.2)',
236                 borderColor: 'rgba(54, 162, 235, 1)',
237                 borderWidth: 1
238             }]},
239         options: {
240             responsive: false,
241             maintainAspectRatio: false,
242             scales: {
243                 y: {beginAtZero: true},
244                 title: {
245                         display: true,
246                         text: 'All values are expressed in [%]'
247                     }},
248             plugins: {
249                 legend: {
250                     display: true,
251                     position: 'top'
252                 }}}
253     });
254     var chart4 = new Chart(ctx4, {
255         type: 'bar',
256         data: {
257             labels: group4Labels,
258             datasets: [{
259                 label: 'OTE (%) indices of the last processing'
                        ,
260                 data: group4Data,
261                 backgroundColor: 'rgba(255, 165, 0, 0.2)',
262                 borderColor: 'rgba(255, 165, 0, 1)',
263                 borderWidth: 1
264             }]},
265         options: {
266             responsive: false,
267             maintainAspectRatio: false,
```

```
268             scales: {
269                 y: {beginAtZero: true},
270                 title: {
271                         display: true,
272                         text: 'All␣values␣are␣expressed␣in␣[%]'
273                     }},
274             plugins: {
275                 legend: {
276                     display: true,
277                     position: 'top'
278                 }}}
279     });
280     # Dati ottenuti dal database
281     var days = <?php echo json_encode($days); ?>;
282     var oeeValues = <?php echo json_encode($oeeValues); ?>;
283     var oteValues = <?php echo json_encode($oteValues); ?>;
284     # Creazione dei line chart
285     var lineChartCTX1 = document.getElementById('lineChart1').
           getContext('2d');
286     var lineChart1 = new Chart(lineChartCTX1, {
287         type: 'line',
288         data: {
289             labels: days,
290             datasets: [{
291                 label: 'OEE_CPFactory␣(%)',
292                 data: oeeValues,
293                 borderColor: 'orange',
294                 fill: false,
295                 # Imposta la tensione per ottenere una forma
                       ondulata
296                 tension: 0.4
297             }]},
298         options: {
299             responsive: false,
300             scales: {
301                 y: {beginAtZero: true,
302                     title: {
303                         display: true,
304                         text: 'All␣values␣are␣expressed␣in␣[%]'
305                     }},
306                 x: {
307                     display: true,
308                     title: {
309                         display: true,
310                         text: 'Days␣of␣the␣selected␣month'
311                     }}}
312         },
```

```
313         plugins: {
314             legend: {
315                 display: true,
316                 position: 'top'
317             }}
318     });
319     var lineChartCTX2 = document.getElementById('lineChart2').
           getContext('2d');
320     var lineChart2 = new Chart(lineChartCTX2, {
321         type: 'line',
322         data: {
323             labels: days,
324             datasets: [{
325                 label: 'OTE_CPFactory (%)',
326                 data: oteValues,
327                 borderColor: 'blue',
328                 fill: false,
329                 tension: 0.4
330             }]},
331         options: {
332             responsive: false,
333             scales: {
334                 y: {beginAtZero: true,
335                     title: {
336                         display: true,
337                         text: 'All values are expressed in [%]'
338                     }},
339                 x: {
340                     display: true,
341                     title: {
342                         display: true,
343                         text: 'Days of the selected month'
344                     }}}
345         },
346         plugins: {
347             legend: {
348                 display: true,
349                 position: 'top'
350             }}
351     });
352 }
353 </script>
354 <?php endif; ?>
355 </body>
356 </html>
```

Inoltre è stato utilizzato un secondo file (`C:\xampp\htdocs\divisioneColonne.css`) necessario per la suddivisione della finestra web in due colonne, per distribuire i grafici a barre sulla sinistra e i grafici a linea sulla destra.

```
1 # Questo file CSS definisce alcune regole per la disposizione
    dei grafici sulla pagina web utilizzando il concetto di
    flexbox
2 .chart-container {
3   display: flex;
4   justify-content: space-between;
5 }
6 .chart-column {
7   flex: 1;
8   margin-right: 20px;
9 }
```

Bibliografia

[1] Schober, Weiss, and Wascher. *MES 4 Manual.* *https://ip.festo-didactic.com/InfoPortal/CPFactoryLab/overview/index.php?lang=en*. Feb. 2019.

[2] Franco Ruvoletto, Mario Beltrame, and Diego Carraro. Doee - kpi, indicatori e misure nella produzione: Oee e ole. `http://www.doeeconsulting.it/indicatori-produzione-ole.aspx`, 2021.

[3] Peter Dobra and János Jósvai. Increase oee at manual assembly lines by data mining. *Acta Technica Jaurinensis*, 13(2):99–111, 2020.

[4] Sebastiano Di Luozzo, S Fiorenza, Mm Schiraldi, et al. On the relationship between human factor and overall equipment effectiveness (oee): evidences from an application of the analytic hierarchy process. In *Proceedings of the COPERMAN2022-Conference on Performance Management*. IT, 2022.

[5] Andrea Bonci, Dorota Stadnicka, and Sauro Longhi. The overall labour effectiveness to improve competitiveness and productivity in human-centered manufacturing. In *International Scientific-Technical Conference MANUFACTURING*, pages 144–155. Springer, 2022.

[6] Massimiliano Schiraldi. *Operations management*. BoD–Books on Demand, 2013.

[7] Thomas W. Hartmann. Mind your metrics. *Quality Progress*, 46(11):64, 11 2013.

[8] Marcello Braglia, Davide Castellano, Marco Frosolini, Mosè Gallo, and Leonardo Marrazzini. Revised overall labour effectiveness, international journal of productivity and performance management. pages 1317–1335, Cham, 2021. Emerald Publishing Limited.

[9] Kanthi Muthiah and Samuel Huang. Overall throughput effectiveness (ote) metric for factory-level performance monitoring and bottleneck detection. *International Journal of Production Research*, 45(20):4753–4769, 2007.