POLITECNICO DI TORINO

Corso di Laurea Magistrale
in Ingegneria Gestionale

Tesi di Laurea Magistrale

Costruzione e Ottimizzazione di un modello di simulazione di un magazzino automatico in ottica Digital Twin

Relatore
Giovanni Zenezini

Candidato
Dario De Luca

Correlatore
Andrea Ferrari

Luglio 2022

Ringraziamenti

Ringrazio il professore Zenezini per l'opportunità di poter partecipare a questo progetto e il correlatore Ferrari che in questi otto mesi di lavoro, ha saputo guidarmi, con suggerimenti pratici, nelle ricerche e nella stesura dell'elaborato.

Ringrazio inoltre Federico e Stefano che si sono rivelati più che colleghi durante questo percorso, ricevendo da loro supporto e spunti per un costante miglioramento.

Indice

Tabella delle figure

Indice delle Tabelle

1. Introduzione

1.1 Introduzione alla Supply Chain e al Supply Chain Management

1.1.1 Definizione di una Supply Chain

La supply chain è il network che include tutti gli individui, le organizzazioni, le risorse, le tecnologie e le attività coinvolte nella creazione e nella vendita di un prodotto: dall'acquisto dei materiali dal fornitore, fino alla consegna del prodotto finito all'utente finale.

In questa definizione, sono elencate una serie di caratteristiche chiave che sono state utilizzate per ritrarre una catena di approvvigionamento. In primo luogo, si forma una supply chain solo se sono presenti più società partecipanti. In secondo luogo, tali società normalmente non appartengono alla stessa proprietà aziendale e quindi esiste un'indipendenza giuridica nel mezzo. In terzo luogo, le società sono interconnesse in base all'impegno comune di aggiungere valore al flusso di materiali che attraversa la catena di approvvigionamento. Questo flusso di materiale, per ciascuna azienda, entra come input ed esce come output di valore aggiunto.

La funzione di collegamento tra fornitore e cliente nella supply chain è svolta dai magazzini. Un magazzino è una struttura per consolidare i prodotti, ridurre i costi di trasporto e i lead time, ottenere economie di scala nella produzione o negli acquisti (Bartholdi III JJ e Hackman ST, 2006), e fornire processi a valore aggiunto (Gong Y e De Koster MBM, 2008). Inoltre, il magazzino è riconosciuto come uno degli asset principali con cui le aziende possono fornire servizi su misura per i propri clienti e ottenere un vantaggio competitivo.

1.1.2 La configurazione di una Supply Chain

In passato, la struttura di una supply chain era di tipo tradizionale, caratterizzata da una configurazione lineare e semplice i cui attori principali erano i fornitori, i produttori, i centri di distribuzione, i magazzini, i rivenditori e, infine, i clienti. Il ruolo svolto da ciascuno di essi è differente e ricopre diverse funzioni. Più specificatamente, il fornitore si occupa di garantire il flusso delle materie prime necessarie all'avvio della produzione, il produttore rappresenta il core della filiera ed ha il compito di realizzare i prodotti finiti e i distributori svolgono la funzione di consolidamento e trasporto della merce verso il retailer che è l'anello di congiunzione con il consumatore finale.

Per quel che riguarda i magazzini, il loro aspetto fondamentale è la gestione del flusso di materiale e le tipiche operazioni sono: ricezione, stoccaggio, rifornimento interno, prelievo degli ordini,

accumulo e smistamento, imballaggio, cross docking e spedizione (Tompkins JA et al., 2003). *(***Figura 1***Figura 1.1*)
Ne esistono diversi tipi che possono essere classificati o come magazzini di produzione e centri di distribuzione (Ghiani G et al., 2004) oppure, in base al loro ruolo nella filiera, come magazzini di materie prime, semi-lavorati o prodotti finiti, centri di distribuzione, magazzini di evasione ordini, depositi locali diretti alla domanda dei clienti e magazzini di servizi a valore aggiunto (Frazelle E, 2001).

Gli effetti di una corretta gestione delle attività di magazzino possono contribuire positivamente sui livelli di servizio offerti al cliente, sui tempi di consegna e sulla struttura dei costi di un'azienda. Quindi si può concludere che il magazzino influenza le prestazioni di un'intera catena di approvvigionamento.

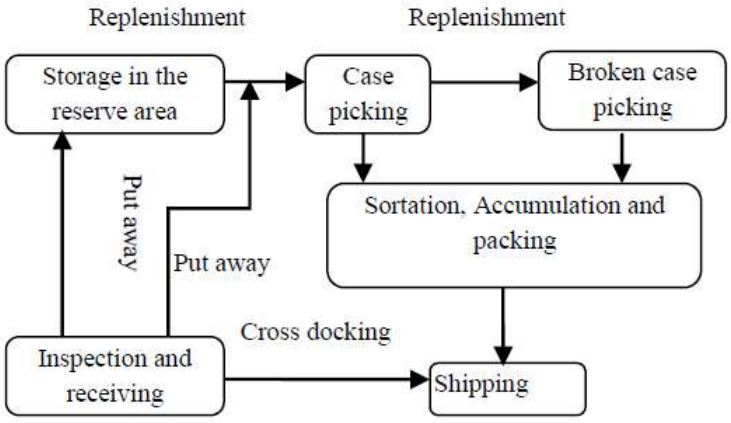


Figura 1.1 *- Operazioni di magazzino (Fonte: Gong, Y., 2009)*

Tuttavia, in tempi recenti, l'avvento della globalizzazione e le nuove tendenze legate ai consumatori e alle modalità di acquisto, quali ad esempio, la vendita su piattaforme e-commerce hanno messo in crisi molti aspetti della struttura tradizionale.
La configurazione e il numero elevato di attori aumentano la complessità delle supply chain (Choi T.Y. et al., 2001) che si trasformano da una forma lineare ad una rete collaborativa; pertanto, sorge la necessità di strumenti di supporto alle decisioni per affrontare meglio le dinamiche nelle supply chain.

Una volta collegate a un network, le aziende diventano sia fornitori che acquirenti delle aziende partecipanti, ciò garantisce loro un'ampia visibilità sulle operazioni interconnesse dei loro partner commerciali. Oltre a consentire alle aziende di identificare più facilmente tendenze o problemi emergenti, la rete consente di collaborare con nuove aziende, migliorare il flusso di cassa, sviluppare nuovi prodotti e accelerare la sostenibilità. Inoltre, permette loro di muoversi rapidamente quando la

domanda del mercato cambia e migliora la resilienza ovvero “la capacità di resistere e di reagire di fronte a difficoltà, avversità, eventi negativi”. (voc. Treccani)

Il consumatore finale rappresenta la domanda da soddisfare e non appartiene alla supply chain ma svolge un ruolo ben differente: è il motore del Supply Chain Management. Dato che il cliente si trova al centro del Supply Chain Management, la filiera si nutre di molte sue informazioni poiché necessarie ad offrirgli un migliore livello di servizio.

1.1.3 Flussi di una supply chain

In una supply chain vi sono quattro flussi principali (*Figura 1.2*):

- Flusso di materiali: il flusso si sviluppa da monte verso valle. Esso è gestito dai magazzini che hanno il ruolo di dirigere il flusso all'interno della rete di distribuzione. Ai diversi stadi di una filiera si possono trovare le materie prime per l’avvio della produzione, i wip che rappresentano i semi-lavorati e, infine, i prodotti finiti. La continuità del flusso dei materiali dovrebbe essere garantita non solo all'interno delle aziende, ma anche nell'intera filiera logistica poiché uno dei fattori fondamentali per la soddisfazione del cliente è la minimizzazione del lead time, ovvero il tempo di attesa;

- Flusso di informazioni: in una catena di approvvigionamento ci sono una moltitudine di flussi di informazioni come il flusso relativo alla domanda, alla previsione, alla pianificazione e alla produzione. A differenza del flusso di materiale, le informazioni possono essere eseguite in entrambe le direzioni, sia verso monte che verso valle. È interessante notare come la maggior parte dei flussi di ogni filiera ha un proprio insieme di informazioni specifiche correlate al proprio business;

- Flusso finanziario: ogni step di una supply chain svolge differenti attività per cui è necessario sostenere un costo. Una delle prospettive più avvalorate individua nel consumatore finale l’unica fonte di finanziamento dell’intera supply chain. La distribuzione e la condivisione equa di questa singola risorsa finanziaria lungo una filiera consente un migliore allineamento tra contributo e compenso per le aziende partecipanti;

- Flusso commerciale: tutta la catena di approvvigionamento rappresenta un flusso commerciale transazionale. Ciò significa che il flusso di materiale che attraversa la supply chain cambia la sua proprietà da un'azienda all'altra. Il processo transazionale di acquisto e vendita sposta ripetutamente la proprietà del flusso di materiale dal fornitore all'acquirente fino a valle della catena, ovvero fino al consumatore finale.

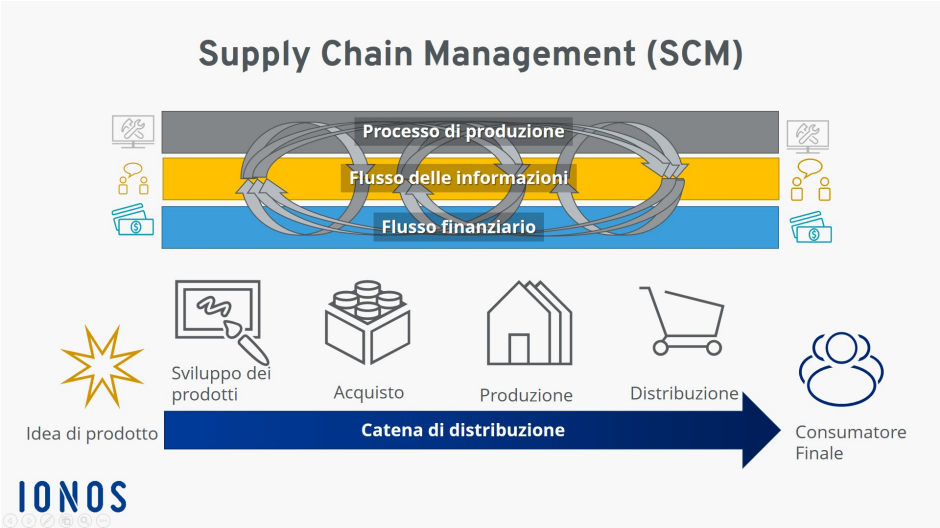


Figura 1.2 *– Principali flussi di una Supply Chain (Fonte: ionos.it)*

1.1.4 Definizione di Supply Chain Management

Da quando nel 1982 Olivier e Webber introdussero per la prima volta il termine Supply Chain Management, numerosi accademici, professionisti e organizzazioni hanno offerto diverse definizioni. Esse sono un insieme svariato di descrizioni e alcune offrono una prospettiva ristretta e basata sul funzionamento, ad esempio: "SCM è la gestione e il controllo di tutti i materiali e le informazioni nel processo logistico, dall'acquisizione delle materie prime alla consegna all'utente finale" (CSX World Terminals, 2004). Altre, invece, definiscono il Supply Chain Management in modo ampio, ad esempio: "SCM è l'integrazione dei processi aziendali dall'utente finale ai fornitori di prodotti, servizi e informazioni che aggiungono valore per i clienti" (Lambert, 1994).

Betchel e Jayaram nel 1997 hanno classificato più di 50 definizioni esistenti in cinque scuole di pensiero e, successivamente, Cooper, Lambert e Pagh hanno fornito una revisione sottolineando la caratteristica secondo cui Supply Chain Management e logistica non sono identiche e il SCM è composto da tre elementi: processi aziendali, gestione dei materiali e struttura della catena di approvvigionamento.

Tuttavia, le definizioni proposte dal 1982 al 2005 non rappresentano il Supply Chain Management in modo coerente. Data la mancanza di consenso riguardo una definizione, alcuni professionisti del settore si sono riuniti nel Council of Supply Chain Management Professionals (CSCMP) per formalizzarla. La stragrande maggioranza degli intervistati ha indicato che il SCM coinvolge sia aspetti di strategia che aspetti operativi. Inoltre, la discussione ha evidenziato gli elementi principali del Supply Chain Management, tra cui le attività di approvvigionamento, di produzione e relative alla logistica. Un'ulteriore caratteristica chiave include la collaborazione con fornitori e clienti.

In definitiva, la definizione ufficiale adottata dal CSCMP è:

“Il Supply Chain Management comprende la pianificazione e la gestione di tutte le attività coinvolte nel reperimento, nell’approvvigionamento, nella trasformazione dei materiali e delle attività della logistica. È importante sottolineare che include anche il coordinamento e la collaborazione tra tutti gli attori della filiera, che possono essere fornitori, intermediari, fornitori di servizi di terze parti e clienti. In sostanza, il SCM integra la gestione della domanda e dell’offerta all’interno e tra le aziende”.

1.1.5 Visioni SCM e Logistica

Un altro aspetto approfondito dalla letteratura relativo al Supply Chain Management riguarda l’interpretazione della logistica; talvolta, quest’ultima è impropriamente utilizzata come sinonimo di Supply Chain Management. Più correttamente, il Supply Chain Management formalizza un modello di business coeso che si sviluppa dal marketing e dalla definizione del prodotto e raggiunge gli aspetti relativi alla finanza e ai clienti; mentre, la logistica verte unicamente sulla movimentazione e sullo stoccaggio delle merci.

Nel 2004 Paul Larson e Arni Halldorsson nell’International Journal of Logistic presentano quattro differenti visioni per spiegare il rapporto tra supply chain management e logistica *(Figura 1.3):*

- *traditionalist:* posiziona il Supply Chain Management all’interno della logistica, ovvero SCM è una piccola parte della logistica;
- *re-labeling:* è una prospettiva che considera la logistica uguale al SCM;
- *unionist:* considera la logistica come parte del Supply Chain Management; come suggerito da Mentzer et al. (2001) “tutte le funzioni aziendali dovrebbero essere incluse nel processo di Supply Chain Management”. Nel loro modello di SCM, queste funzioni aziendali tradizionali sono: marketing e vendite, ricerca e sviluppo, previsioni, produzione, acquisti, logistica, sistemi informativi, finanza e servizio clienti;
- *intersectionist:* il concetto di “intersezione” suggerisce che Supply Chain Management non è l’unione di logistica, marketing, gestione delle operazioni, acquisti e altre aree funzionali, piuttosto, include elementi strategici e integrativi di tutte queste discipline. Tale visione è stata supportata da Giunipero e Brand che nel 1996 affermano: “SCM non è un sottoinsieme della logistica ma è una strategia ampia che attraversa i processi aziendali sia all’interno dell’azienda che attraverso i canali esterni”.

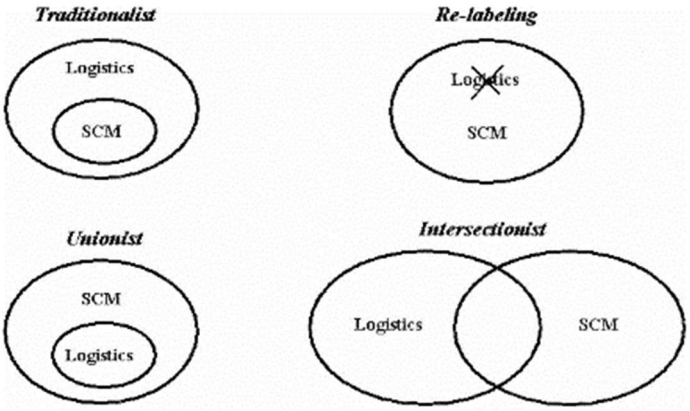


Figura 1.3 – *Visioni SCM-Logistica (Fonte: Mason, R.J., 2009)*

Fra le quattro visioni proposte, il CSCMP ha sostenuto la visione "unionista" ed ha modificato la definizione di logistica. Il Concilio ritiene che: "La logistica è quella parte del Supply Chain Management che pianifica, implementa e controlla il flusso lungo la filiera in modo che sia efficiente ed efficace; inoltre, comprende lo stoccaggio di beni, la fornitura di servizi e la condivisione delle informazioni al fine di soddisfare i requisiti dei clienti".

1.1.6 Processi del Supply Chain Management

L'ambiente competitivo che è emerso in tempi recenti ha generato un significativo cambiamento di paradigma nella gestione aziendale per cui le imprese non competono più come entità autonome bensì si aggregano in supply chain. Logicamente, la gestione di un insieme di aziende risulta più difficile rispetto ad una singola. Come affermato da Tom Blackstock, ex vicepresidente delle *operations* di Coca Cola Nord America, è richiesto il coinvolgimento di tutte le funzioni aziendali a supporto del Supply Chain Management per far fronte all'aumento di complessità.

Come già visto, è improprio riferirsi ad una catena quanto più ad una rete di società che si relaziona attraverso processi intra- e inter- aziendali appartenenti al Supply Chain Management. In una rete, ogni organizzazione deve implementare dei processi di business uniformati in modo tale da rendere le transazioni efficaci ed efficienti e strutturare le relazioni tra le aziende. Tutti i processi coinvolgono le funzioni aziendali svolte in ciascun livello della filiera, fra cui le principali sono: acquisti, produzione, logistica, ricerca e sviluppo, marketing e finanza.

È necessario che questi processi siano innovati. La riprogettazione si basa sulla standardizzazione sia delle transazioni sia del trasferimento delle informazioni; ne deriva che condurre transazioni multiple con lo stesso cliente è più efficiente che ottenerne uno nuovo. L'obiettivo è quello di migliorare le transazioni con i clienti in quanto una loro maggiore soddisfazione aumenta la possibilità di consolidare il rapporto.

Nel 1998, il Global Supply Chain Forum (GSCF) ha fornito le linee guida per descrivere gli otto processi principali. Ogni processo di SCM ha dei sottoprocessi strategici e operativi. I primi forniscono la struttura di implementazione del processo, i secondi forniscono i passi dettagliati per l'esecuzione. Il processo strategico è un passo necessario per integrare l'azienda con altri membri della supply chain mentre le attività quotidiane sono svolte a livello operativo. Ogni processo è guidato da un team composto da manager di ciascuna funzione aziendale; a livello strategico i team sono responsabili dello sviluppo delle procedure e, contestualmente, a livello operativo si occupano di gestire la loro implementazione.

I processi di gestione della catena di fornitura identificati dal Global Supply Chain Forum sono:

- *Customer Relationship Management:* la gestione identifica i clienti chiave e i gruppi di clienti da considerare come parte della missione commerciale dell'azienda. L'obiettivo è quello di segmentarli in base al loro valore nel tempo e aumentare la loro fedeltà fornendo prodotti e servizi personalizzati. I team lavorano con i clienti chiave per migliorare i processi e ridurre le attività non a valore aggiunto. Il sistema genera dei rapporti sulle prestazioni per misurare la redditività dei singoli clienti così come l'impatto dell'azienda sulla performance finanziaria dell'acquirente;

- *Supplier Relationship Management:* come il nome suggerisce, questo processo è molto simile al Customer Relationship Management. Proprio come un'azienda ha bisogno di sviluppare relazioni strette con i suoi clienti principali, ha anche bisogno di promuovere tali relazioni con i suoi fornitori chiave. Le partnership sono sviluppate con un piccolo sottoinsieme di fornitori in base al valore che apportano all'organizzazione nel tempo, mentre relazioni più tradizionali sono mantenute con gli altri. Il risultato desiderato è una relazione in cui entrambe le parti ne beneficiano;

- *Customer Service Management:* è il processo di SCM che si occupa dell'amministrazione degli accordi sviluppati dai team dei clienti durante il processo di Customer Relationship Management. I manager di questo processo controllano gli accordi e intervengono proattivamente per conto del cliente in caso di un problema nel rispetto delle promesse fatte. L'obiettivo è quello di risolvere i problemi prima che incidano sul cliente. I manager del Customer Service Management si interfacciano con gli altri team di processo per assicurare che gli accordi siano rispettati;

- *Demand Management:* è il processo di SCM che equilibra la domanda dei clienti con le capacità della catena di approvvigionamento. Con il giusto processo in atto, la gestione può far coincidere l'offerta con la domanda in modo proattivo ed eseguire il piano con interruzioni minime. Il processo non è limitato alla previsione e alla sincronizzazione della domanda con l'offerta, bensì comprende anche la riduzione della variabilità e l'aumento della flessibilità.

Una buona gestione della domanda utilizza i dati dei punti vendita e dei clienti chiave per ridurre l'incertezza e fornire flussi efficienti in tutta la catena;

- *Order Fulfillment:* il processo di evasione degli ordini include tutte le attività necessarie per progettare una rete e permettere a un'azienda di soddisfare le richieste dei clienti massimizzando la sua redditività. Gran parte del lavoro effettivo sarà eseguito dalla funzione logistica, nonostante ciò, il processo deve essere implementato a livello inter-funzionale. L'obiettivo è quello di sviluppare un processo senza interruzione di continuità che comprende sia i differenti segmenti dei clienti sia i fornitori;

- *Manufacturing Flow Management:* è il processo di gestione della supply chain che include tutte le attività necessarie per ottenere, implementare e gestire la flessibilità di produzione nella filiera e per movimentare i prodotti. Tale flessibilità riflette la capacità di produrre un'ampia varietà di prodotti in modo tempestivo e al minor costo possibile;

- *Product Development and Commercialization:* è il processo che fornisce la struttura per sviluppare e immettere il prodotto sul mercato operando in collaborazione con clienti e fornitori. Il team di Product Development and Commercialization deve coordinarsi con il team del Customer Relationship Management per identificare le esigenze esposte dal cliente, con quello del Supplier Relationship Management per selezionare i materiali e i fornitori e, infine, con quello del Manufacturing Flow Management per sviluppare la tecnologia di produzione e implementare il miglior flusso di prodotto per la combinazione prodotto/mercato;

- *Returns Management:* è il processo che comprende le attività associate ai resi e alla logistica inversa, ed è gestito sia all'interno dell'azienda che tra gli attori chiave della supply chain. La corretta implementazione di questo processo permette al management non solo di gestire il flusso inverso dei prodotti in modo efficiente, ma anche di identificare le opportunità per evitare i resi e i resi indesiderati. Attraverso una migliore gestione della logistica inversa è possibile ottenere una riduzione dei costi, tuttavia, evitando quelle pratiche di gestione e quei fallimenti di performance che causano i resi, la riduzione è ancora più significativa.

1.2 L'Industria 4.0

Il termine *Industry 4.0* è stato utilizzato per la prima volta nel 2011, alla fiera di Hannover, in Germania, per indicare un progetto rivoluzionario per il settore industriale. Il focus del progetto è relativo ad un mix di tecnologie che permettono la compenetrazione fra il mondo fisico, biologico e digitale con l'obbiettivo di trasformare le aziende in *smart factory*, ovvero organizzazioni più complesse caratterizzate da automazione, digitalizzazione, connessione e programmazione.

Le ragioni di rinnovare il settore e l'approccio al lavoro ricadono nella necessità di far fronte all'aumento di complessità nella struttura delle supply chain causato dalla globalizzazione e dal grado di interconnessione esponenzialmente crescente tra i *players*, alla maggiore complessità dei prodotti e delle loro specificità, e, infine, all'incremento della mole di dati da gestire correttamente e da conservare. La direzione di questa rivoluzione è indirizzata al miglioramento della produttività, delle condizioni di lavoro e allo sviluppo di nuovi processi.

Le tecnologie introdotte con l'Industria 4.0 sono fondamentali per innovare i processi, i prodotti e i servizi in tutti i settori; inoltre, servono a creare un collegamento fra la realtà fisica e quella virtuale.

Una delle tecnologie più importanti è quella dell'Internet of Things (IoT); essa permette di connettere oggetti, dispositivi, macchinari, robot e lavoratori con Internet, purché dotati di identità digitale, oltre che reale. L'utilità dell'IoT non riguarda solo la connessione delle entità bensì la capacità di acquisire dati utili a modificarne il funzionamento in ottica di miglioramento continuo.
L'applicazione ai magazzini di un sistema logistico intelligente basato sull'IoT permette di avere una comprensione più intuitiva e accurata dell'inventario corrente e ampia tracciabilità in tempo reale della merce, sia durante le operazioni interne di magazzino, sia nelle fasi di trasporto.

In secondo luogo, nel processo di produzione sono inseriti i robot collaborativi capaci di condividere lo spazio di lavoro con operatori o altri robot in modo efficiente e sicuro grazie all'Intelligenza Artificiale (IA). L'utilizzo di robot in contesti produttivi può sembrare un argomento noto e superato; tuttavia, nelle circostanze descritte si intendono robot migliorati ed evoluti, in grado di essere autosufficienti, autonomi, e interattivi *(Figura 1.4* - Robot MIR*)*. Le occupazioni riservate ai robot sono soprattutto per lavori ripetitivi e standardizzabili.

Figura 1.4 *- Robot MIR (Fonte: mobile-industrial-robots.com)*

In ambito di Additive Manufacturing sono state introdotte stampanti 3D connesse a software di sviluppo digitale. Tali strumenti permettono di creare oggetti fisici tridimensionali partendo da un

modello digitale che ne identifica le caratteristiche progettuali. In questo caso, l'approdo di nuove tecnologie, ha rivoluzionato il modo di produrre, infatti la manifattura tradizionale, fondata sull'asportazione di materiale, è sostituita dalla manifattura additiva che, al contrario, funziona per depositi stratificati di materiale. Questo cambio di paradigma ha reso possibile non solo un minore utilizzo di materia prima e riduzione dei costi, ma, in particolar modo, la realizzazione di prodotti con forme geometriche complesse, oltre che personalizzate. La stampa in 3D non è effettivamente così recente come innovazione, ma solo negli ultimi tempi il settore è stato abbastanza maturo da poterla sfruttare adeguatamente.

È importante comprendere come uno degli aspetti centrali apportati dalla trasformazione digitale sia l'attenzione sui Big Data; questo termine si usa per esprimere una grossa mole di informazioni aggregate che, sono raccolte, come già visto, dai robot e da tutti i dispositivi connessi alla rete. Un binomio indissolubile che si affianca al concetto di Big Data è quello di Analytics con cui si indicano i modelli di analisi utilizzati per interpretare tutti i dati. Qualora fossero svolte delle analisi intelligenti di tipo qualitativo sui dati, con il fine di aggiungere valore alle informazioni, allora si parlerebbe, non solo di Big Data, bensì di Smart Data. I benefici relativi alle analisi dei dati aziendali hanno garantito l'aumento della produzione e il rispettivo abbassamento dei costi legati ad essa, oltre che ottenere maggiore flessibilità ed efficienza. Lo studio dei dati è eseguito anche attraverso simulazioni e previsioni a breve e lungo termine, creando la possibilità di analizzare preventivamente differenti scenari in un tempo e ad un costo minore rispetto al passato. Le simulazioni sono eseguite in tempo reale, con dati reali, in modelli virtuali controllati, in modo da testare e ottimizzare i processi aziendali ancora prima della loro realizzazione fisica. La simulazione è uno strumento molto potente che permette di evitare gli ingenti costi derivanti dal *learning-by-doing* oltre che migliorare la qualità e abbassare i tempi e i costi. Anticipando il futuro, le imprese tentano di ridurre i rischi aziendali e di ottenere un vantaggio competitivo sul mercato.

È evidente che una tecnologia come il Cloud sia fondamentale in un contesto come quello appena descritto; infatti, il Cloud rappresenta un'infrastruttura IT flessibile e aperta alla condivisione dei dati attraverso la rete Internet. In questo modo i dati possono essere condivisi anche all'esterno dei confini aziendali in modo tale da poter seguire la trasformazione dei modelli di business o di realizzare delle collaborazioni con altre imprese.

Infine, vi è l'aspetto della sicurezza delle informazioni rilevanti per il business, a tal proposito si parla di cybersecurity. Con essa si fa riferimento all'uso di protocolli standardizzati per la protezione dei dati; tali attenzioni sono necessarie affinché le comunicazioni siano sicure ed affidabili, non ci siano minacce informatiche per le aziende e, in ultimo, sia garantita la sicurezza nella gestione delle identità delle macchine e degli utenti sulla rete.

Le tecnologie che sono state introdotte dall'avvento della Quarta Rivoluzione Industriale non si esauriscono con quelle descritte finora, bensì ne esistono altre relative ad aspetti differenti. In particolar modo tra le più rilevanti si può indicare la blockchain.

Con il termine di blockchain si indica un registro condiviso e immutabile che si integra con il processo di registrazione delle transazioni commerciali, favorendone la tracciabilità. Gli aspetti positivi connessi alla blockchain riguardano l'aumento di trasparenza delle informazioni trasmesse, nonché immediatezza e possibilità di condivisione dei dati; oltre a ciò, vi è una riduzione dei rischi legati agli scambi e ai relativi costi, in più è possibile svolgere transazioni con qualsiasi tipo di asset, tangibile o intangibile che sia.

La direzione presa dell'industria mondiale ha comportato l'aumento di velocità dei tempi di finalizzazione dei prodotti, maggiore qualità degli stessi, meno margine di errore e una nuova modalità di business caratterizzata da una visione meno legata alla vicinanza territoriale e più globalizzata.

Tali cambiamenti hanno riguardato sia le grandi sia le medio-piccole imprese; le prime ricoprono il ruolo di leadership, mentre le seconde possono affacciarsi su un panorama mondiale grazie agli aiuti, alle tutele e alle capacità sviluppate nel tempo. Un ulteriore cambiamento è l'aumento della competitività: ora si fa business in tutto il mondo "contro" tutte le altre aziende.

Per quanto riguarda il nostro continente, la Commissione Europea e i governi degli stati appartenenti, fra cui quello Italiano, hanno promosso e finanziato un piano volto alla creazione di un mercato unico digitale. Nel periodo 2014-2020 si è sviluppato il progetto chiamato *Horizon 2020* che ha portato all'Europa l'eccellenza scientifica. Successivamente, per gli anni 2021-2027, la Commissione vuole sviluppare un programma, detto *Horizon Europe*, stanziando 100 miliardi di euro per le imprese degli stati europei.

L'ambiente aziendale ha subito una trasformazione significativa poiché l'adozione dell'Industria 4.0 nel settore manifatturiero ha rivoluzionato i processi e le operazioni di produzione (DeSousaJabbour et al., 2018) e ciò ha portato a un progresso nei sistemi di produzione (Li, 2018). Inoltre, è stato possibile migliorare gli aspetti riguardanti la sostenibilità ambientale e sociale come il consumo di energia, di acqua, di emissioni di gas serra oltre che la sicurezza e la salute dei lavoratori. In altri livelli della filiera, ad esempio i distributori, tali miglioramenti sono stati di minore entità poiché hanno coinvolto meno attività.

Esistono anche possibili aspetti negativi legati alle tecnologie dell'Industria 4.0 come l'incremento della disuguaglianza che si potrebbe verificare nel mercato del lavoro, portando ad una fossilizzazione sui ruoli "bassa competenza-basso stipendio" e "alta competenza-alto stipendio". Un'altra preoccupazione riguarda la possibile sostituzione dei lavoratori in favore dei robot in molte delle attuali mansioni, soprattutto quelle operative. Tuttavia, sebbene si sia parlato di nuove tecnologie e

innovazioni, l'attore principale di una azienda rimane sempre, e comunque, l'uomo, qualsiasi sia il contesto.

Tutto questo si pone lo scopo di rendere più efficienti i processi, migliorare la qualità e diminuire i costi. Grazie alla capacità di trasmissione dei dati in tempo reale e con precisione, è possibile migliorare le performance e la coordinazione di tutta la supply chain. Altri benefici potenzialmente derivanti dalle tecnologie 4.0 sono l'incremento della produttività, miglior utilizzo delle risorse, della collaborazione orizzontale e verticale tra gli attori della filiera, migliore rapporto con i clienti, maggiore flessibilità ai cambiamenti di mercato, diminuzione del costo relativo allo sviluppo dei prodotti e minori lead time.

Tuttavia, l'attenzione delle imprese non dovrebbe concentrarsi unicamente sugli obiettivi di riduzione dei costi o dei tempi delle attività, bensì, anche sulla sostenibilità ambientale.
Secondo Dyllick e Hockerts (2002) e Veleva ed Ellenbecker (2001), la sostenibilità aziendale può essere definita come il raggiungimento della prosperità economica attraverso la creazione di prodotti e servizi redditizi evitando la generazione di impatti negativi sull'ambiente e sulla società. L'interesse per la sostenibilità è aumentato in molti settori e sono stati sviluppati vari indicatori per valutare e rendicontare le prestazioni economiche, ambientali e sociali delle imprese (Slaper e Hall, 2011).

La domanda di prodotti e servizi appartenenti ad aziende e supply chain indirizzate alla sostenibilità è in forte crescita, sottolineando come questo aspetto è considerato un vantaggio competitivo sul mercato.

1.2.1 Logistica 4.0

La logistica comprende varie funzioni, tra cui il trasporto, la distribuzione, la gestione del magazzino e dell'inventario, nonché la logistica inversa (Aguezzoul, 2014). Negli ultimi anni si è assistito alla crescente integrazione nella logistica di Cyber Physical Systems (CPS), IoT e altri sistemi basati sull'intelligenza artificiale (IA) secondo la tendenza dell'Industria 4.0, portando alla recente introduzione del termine "Logistica 4.0" o "Logistica Intelligente".

Si parla di Logistica 4.0 quando ci si riferisce alle attività svolte attraverso le tecnologie basate su Internet per migliorare l'accuratezza, l'efficienza, la visibilità, la reattività e la flessibilità lungo l'intera catena del valore (Prause, 2015); in tal modo le supply chain diventano più innovative, competitive e sostenibili. Per raggiungere questi obiettivi, non è sufficiente istituire processi di produzione intelligenti o automatizzati, ma devono essere implementati anche altri processi lungo la catena di approvvigionamento, comprese le operazioni logistiche (Yu e Solvang, 2017).

Barretto et al. (2017) definiscono la Logistica 4.0 come l'ottimizzazione delle attività logistiche attraverso l'uso di sistemi intelligenti, incorporati in database o software, in cui tutti i processi rilevanti possono comunicare tra loro e con gli esseri umani al fine di migliorare le capacità analitiche e operative. La comunicazione tra i processi rilevanti è talvolta chiamata "decentramento" o "autoregolamentazione" (Hofmann e Rusch, 2017).

Hofmann e Rusch, (2017), Rakyta et al. (2016) e Bag et al. (2020) citano molteplici esempi di iniziative di Logistica 4.0; in primo luogo, i sistemi di gestione del magazzino (WMS) e il monitoraggio in tempo reale dei flussi di materiale attraverso la tecnologia di identificazione automatica, i tag di identificazione a radiofrequenza (RFID) e i dispositivi di scansione. In secondo luogo, applicazioni per l'automazione della movimentazione e dello stoccaggio del materiale usufruendo di robot di carico, scarico e prelievo, di veicoli a guida laser (LGV) o a guida automatizzata (AGV), di sistemi di stoccaggio e prelievo automatizzati (AS/RS). Infine, processi di elaborazione autonoma degli ordini e monitoraggio e pianificazione in tempo reale della distribuzione dei prodotti, tramite sistemi di trasporto intelligenti (ITS) e un'unità di controllo telematica (TCU), per acquisire dati rilevanti, tracciare la posizione delle merci e consigliare un percorso ottimale ai conducenti.

Secondo la letteratura, la Logistica 4.0 promuove una maggiore trasparenza o visibilità della supply chain, aumenta la tracciabilità di materiali e prodotti lungo la filiera, migliora la capacità di controllo dell'integrità, il servizio clienti e l'efficienza operativa e decisionale, infine, riduce i costi di stoccaggio e funzionamento. Seyedghorban e Samson, (2020), Bag et al. (2020) e Lorentz et al. (2021) rivelano che l'adozione delle tecnologie digitali aiuta le aziende a modernizzare i propri processi di approvvigionamento, analisi dei dati e pianificazione strategica.

Ad esempio, la tecnologia di identificazione automatica RFID può essere utilizzata per tracciare il flusso dei materiali in tempo reale, inviare notifiche immediate ai partner della catena di approvvigionamento e regolare la produzione in base alla domanda attuale. Questi sforzi migliorano l'efficacia e l'efficienza dei sistemi di produzione *just-in-time (JIT)* e *pull*. Il cloud computing e l'analisi dei big data consentono di identificare i modelli di mercato e migliorare i processi di comunicazione e transazione, particolarmente significativi per i sistemi logistici globali.

Tuttavia, le aziende devono riconoscere che i sistemi basati su Internet e sull'intelligence richiedono necessariamente un'infrastruttura di rete di comunicazione affidabile e ad alta velocità (Thoben et al., 2017), che, a sua volta, richiede costi di investimento elevati. (Maslari C. et al., 2016) notano che per ottenere un'integrazione più efficiente tra i sistemi logistici e internet, è necessario effettuare investimenti significativi in strutture di movimentazione dei materiali, risorse di trasporto e sistemi basati sull'intelligence. Uno studio empirico di Nimawat e Gidwani (2021) conclude che gli elevati costi di investimento rappresentano una barriera critica per i produttori che perseguono l'Industria 4.0.

1.2.2 Warehousing 4.0

Il magazzino è considerato una parte necessaria della supply chain che mira al funzionamento ottimale e continuo dei processi di produzione e distribuzione. Le chiavi principali che garantiscono prestazioni di successo del magazzino sono la progettazione, il layout e le attività svolte (Khojasteh-Ghamari Y., 2012). Un nuovo ruolo per il magazzino è rendere i processi più integrati in tutta la catena di approvvigionamento e fornire visibilità per evitare un livello elevato di inventario che comporta maggiori costi alle aziende (Dowlatshahi S., 2012). Il coordinamento e la condivisione delle informazioni lungo tutta la filiera è l'obiettivo principale che tutte le imprese cercano di raggiungere, dato che questa è la prima arma di difesa contro l'effetto *bullwhip*. Pertanto, la condivisione delle informazioni nella filiera è un problema critico, in particolare le informazioni sull'inventario. Per tali ragioni, si è giunti alla necessità di migliorare le tecnologie all'interno dei magazzini per renderli dei Magazzini 4.0.

Per *Magazzino 4.0* si intende un nuovo modo di movimentare la merce, sfruttando un sistema costituito da una navetta (*shuttle*) mobile al posto di una gru fissa (*fixed crane*) *(Figura 1.5* – Shuttle in Warehousing 4.0 (Fonte: scaffsystem.it). Sia chiaro, si parla sempre di magazzini automatici, tuttavia è diverso il funzionamento. Il processo relativo allo sviluppo del concetto di Warehousing 4.0 richiede la gestione simultanea sia delle infrastrutture di produzione sia quelle di immagazzinamento, oltre che le tecnologie di trasporto e i sistemi di gestione del magazzino.

Una possibile applicazione che ricopre quanto detto finora è il sistema *Automated and Retrieval System* (*AS/RS*), ovvero il sistema di recupero e stoccaggio mediante uno shuttle, detto anche navetta o elevatore. All'interno di questa struttura si svolgono due tipi di missioni: quelle di stoccaggio (*storage mission*) e quelle di recupero (*retrieval mission*); le prime hanno lo scopo di rifornire il magazzino di materiale, al contrario, le missioni di recupero servono a prelevare il materiale e renderlo disponibile all'operatore; le operazioni che la macchina deve svolgere sono simili ma sono eseguite in maniera opposta.

Gli aspetti fondamentali per la descrizione di un Magazzino 4.0 riguardano le dimensioni di lunghezza e altezza delle scaffalature, le velocità, le accelerazioni/decelerazioni e i tempi di carico/scarico della navetta. La progettazione di una specifica scaffalatura deve tenere in conto di molti aspetti, fra cui il genere di prodotti che devono essere stoccati, i relativi flussi in input e output e le misure di performance che si vogliono ottenere, come ad esempio determinati valori di *throughput* e *cycle time*.

È possibile condurre degli studi a priori che, prendendo in input le informazioni relative ad un magazzino, trovano in output i valori attesi delle misure di performance e, inoltre, individuano gli scenari in cui il magazzino risulta più efficiente.

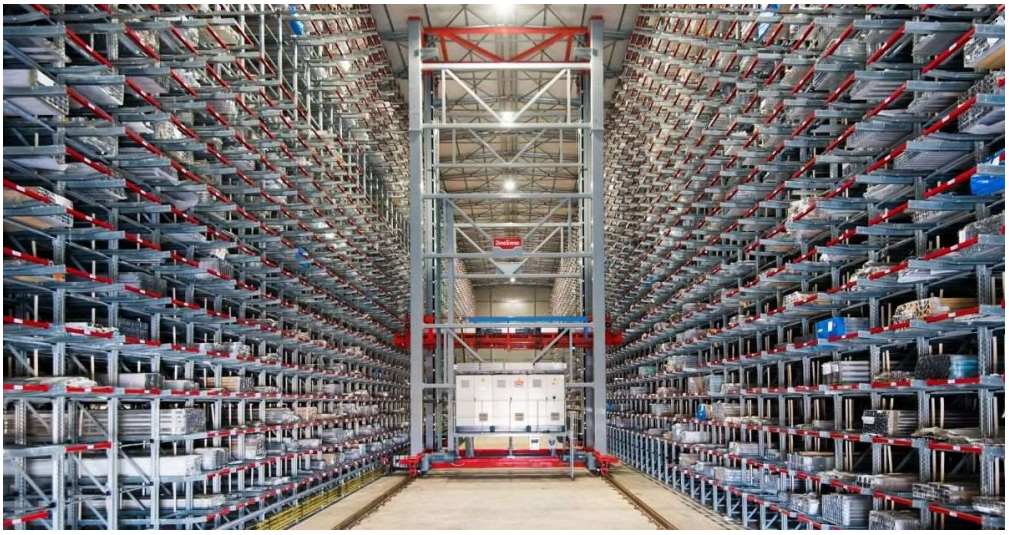


Figura 1.5 *– Shuttle in Warehousing 4.0 (Fonte: scaffsystem.it)*

1.3 Digital Shadow & Digital Twin

L'elemento principale degli impianti industriali più innovativi sono i dati, di conseguenza il loro volume e la loro complessità aumenta. Attraverso le tecniche di Big Data Analytics le aziende sono in grado di elaborare grandi moli di dati al fine di renderli utilizzabili per ottimizzare i processi produttivi, innovare la gestione dei flussi e migliorare gli aspetti di analisi aziendali. Inoltre, questi dati sono utilizzati in laboratori virtuali per la simulazione del comportamento e delle interazioni tra macchine e macchine-dipendenti, garantendo la possibilità di condurre previsioni realistiche e più vantaggiose sia in termini di tempo che di costo rispetto alla simulazione reale.

Due strumenti sviluppati dall'Industria 4.0 che sfruttano la capacità di apprendere e gestire i dati sono il Digital Shadow (DS) e il Digital Twin (DT). Essi modellizzano un sistema fisico, caratterizzato da proprietà descrivibili e soggette a cambiamenti di stato legati al processo, in un sistema virtuale.

Dividendo il discorso, il Digital Shadow permette un flusso di dati unidirezionale dal sistema reale al modello virtuale (Errandonea I. et al., 2020). *(Figura 1.6 –)*
Il DS pone le basi per lo sviluppo del DT attraverso la misurazione di dati e metadati relativi a specifici oggetti che abbiano un riferimento spaziale o temporale, tuttavia, non arriva a descrivere fisicamente l'oggetto e le sue proprietà. Pertanto, è da intendersi come una fase preliminare di un DT poiché è una pura assegnazione delle variabili di stato. Dal punto di vista scientifico, non si genera alcuna conoscenza innovativa, ma costituisce le fondamenta per l'analisi statistica tra differenti categorie di agenti o fasi del processo.

Per quanto riguarda il Digital Twin, esso è stato applicato in diversi contesti industriali, come, ad esempio, per la previsione della fatica e dei danni degli aerei (Tuegel et al., 2011) o per supportare i sistemi di produzione cyber-fisici (Ding et al., 2019). Tuttavia, la letteratura sui concetti di DT basati sulla simulazione applicati ai processi industriali, come per il magazzino automatizzato, è ancora scarsa (Coelho et al., 2021).
Il Digital Twin di un sistema è una rappresentazione virtuale del suo stato fisico. I cambiamenti di stato sono descritti da modelli e supportati da dati. Un DT si distingue da un DS, in quanto permette lo scambio bidirezionale automatizzato di dati tra modello fisico e virtuale e l'autogestione ottimizzata attraverso la sincronizzazione dei dati in tempo reale (Kritzinger et al., 2018) *(Figura 1.6 –)*. La struttura e la realizzazione del DT devono essere orientate all'applicazione e non sono universalmente valide. Il DT dovrebbe anche essere in grado di mappare la logica e le regole utilizzate nel processo o nel comportamento delle entità fisiche (Chen G. et al., 2020); inoltre, dovrebbe codificare i dati e riflettere le previsioni passate, attuali e future dell'entità fisica, inclusi gli oggetti e i processi.

In anni più recenti diversi studiosi hanno provato ad implementare dei DT per i magazzini andando ad analizzare la trasmissione dati attraverso i sensori RFID, l'ottimizzazione del flusso e del consumo elettrico, l'integrazione con strumenti di supporto alle decisioni o per i modelli di *Machine Learning*.

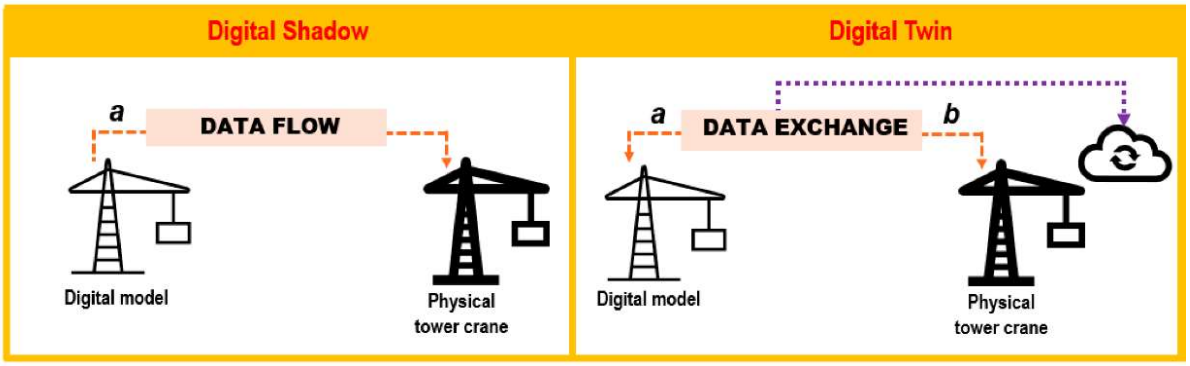


Figura 1.6 – *Flusso di dati unidirezionale e bidirezionale per distinguere DS da DT (Fonte: Sepasgozar, S. M., 2021)*

1.4 Obiettivo della ricerca e struttura della tesi

Con l'avvento di tecnologie all'avanguardia, la modellazione di simulazione ha assunto un ruolo rilevante per analizzare e migliorare i processi aziendali. L'obiettivo di questa ricerca è mostrare un esempio di come le teorie e i principi della modellazione della simulazione possono essere applicati a un caso di studio. In particolare, il sistema in analisi è costituito da un magazzino automatizzato caratterizzato dalla movimentazione dei materiali attraverso tecnologie innovative come un trasloelevatore mini-load e un'infrastruttura gestionale all'avanguardia.

Si tratta di un progetto proposto dal Dipartimento di Ingegneria Gestionale e della Produzione del Politecnico di Torino, che prevede di realizzare un modello virtuale del magazzino presente presso il Laboratorio del DIGEP. Lo sviluppo è avvenuto mediante il software di simulazione AnyLogic, al fine di valutarne i comportamenti e sviluppare un algoritmo ed il rispettivo pseudo-codice per l'ottimizzazione del processo di prelievo e stoccaggio.

La metodologia di ricerca si compone di diverse fasi. Inizialmente si è svolta una fase di studio della letteratura che riguarda i principi generali delle Supply Chain e del Supply Chain Management uniti ai concetti di Industria 4.0 analizzati nell'introduzione, e le teorie alla base della modellazione simulata. Con queste conoscenze di base, la ricerca è proseguita con l'analisi del sistema reale al fine di identificare gli elementi fondamentali da considerare nella costruzione del modello. Successivamente, è seguita la fase di costruzione di un modello concettuale (UML), che consiste nella descrizione di tutti gli elementi che caratterizzano il modello, comprese tutte le assunzioni e semplificazioni del processo di astrazione. Infine, è stato tradotto il modello concettuale in linguaggio informatico su AnyLogic. Per fare ciò, è stato condotto uno studio approfondito del software e delle sue potenzialità. Alla costruzione del modello virtuale è seguito poi il processo di verifica e lo sviluppo di un algoritmo di ottimizzazione dell'attuale funzionamento del magazzino, oltre che un'analisi dei risultati ottenuti dalle simulazioni.

La tesi è composta da dieci capitoli così suddivisi: il primo capitolo, appena descritto, introduce l'ambito in cui la tesi è descritta, mostrando i concetti di supply chain, supply chain management, industria 4.0 e digital twin/digital shadow. Il secondo capitolo contiene un'analisi della letteratura sui magazzini approfondendo le tipologie e gli aspetti legati agli automated storage and retrieval system. Il terzo capitolo presenta i concetti della simulazione e i modelli agent-based e discrete event simulation. Il quarto capitolo mostra una panoramica sulla metodologia utilizzata nel lavoro di tesi. Il quinto capitolo presenta il sistema fisico del laboratorio e comprende una descrizione del suo funzionamento. Il sesto capitolo illustra il modello concettuale con tutti gli elementi inclusi nel modello e le assunzioni effettuate. Il settimo capitolo descrive le caratteristiche del modello di simulazione realizzato mediante il software Anylogic. L'ottavo capitolo contiene l'algoritmo di ottimizzazione e il relativo pseudo-codice. Infine, l'ultimo capitolo, illustra i risultati del lavoro di tesi e le conclusioni.

2. I Magazzini

Il magazzino è una struttura logistica che consente di regolare i flussi di materiale e permette alle aziende di ricevere, conservare e distribuire la merce.
Tale struttura è una parte fondamentale di una supply chain in quanto ha un impatto diretto sui costi di inventario e sui tempi di consegna (Li et al., 2018); inoltre, si stima che il magazzino rappresenti circa il 25% del costo totale della logistica (Tompkins et al., 2015).
Tuttavia, un magazzino efficiente può portare ad una maggiore produttività dell'intera filiera (Drakaki e Tzionas, 2016) con la conseguente diminuzione del costo totale.

La progettazione del magazzino è rilevante per garantire l'efficienza in quanto i costi sono in larga misura determinati in questa fase (Rouwenhorst et al., 2000). È emerso che la progettazione del magazzino è un processo complesso a causa delle numerose decisioni necessarie per determinare la configurazione fisica e le politiche di controllo (McGinnis et al., 2010).

2.1 Ruolo dei Magazzini

I magazzini sono un aspetto chiave delle moderne supply chain e svolgono un ruolo vitale nel successo, o nel fallimento, delle aziende di oggi (Frazelle, 2002). Sebbene molte di esse abbiano esaminato la possibilità di rifornire il cliente direttamente quando esprime la domanda, ci sono ancora molte circostanze in cui questo non è appropriato. Ciò è dovuto al fatto che i tempi di consegna dei fornitori non possono essere ridotti, in modo economico, ai brevi tempi di consegna richiesti dai clienti; quindi, devono essere serviti dalle scorte piuttosto che dall'ordine (Harrison e van Hoek, 2005). Allo stesso modo, può essere vantaggioso tenere scorte strategiche nei punti di disaccoppiamento della catena per separare le attività di produzione *lean*, caratterizzate da un flusso regolare, dai mercati volatili a valle che richiedono una risposta agile (Christopher e Towill, 2001).
In alternativa, le reti di fornitura e distribuzione possono essere sufficientemente complesse da rendere necessario il consolidamento delle merci nei punti di stoccaggio delle scorte in modo che gli ordini multiprodotto per i clienti possano essere consegnati insieme; ad esempio, presso centri di consolidamento *break-bulk* o *make-bulk* (Higginson e Rilegatore, 2005). I primi di questi centri, si occupano di dividere la merce in pallet di minori quantità e i secondi, al contrario, realizzano pallet multiprodotto di maggiore quantità. Le operazioni di tali magazzini sono fondamentali per fornire elevati livelli di servizio al cliente. Un'ampia percentuale di magazzini offre ai clienti un tempo di consegna di massimo un giorno in presenza di inventario (Baker, 2004); il raggiungimento di questo obiettivo necessita affidabilità entro livelli di tolleranza di velocità, precisione e assenza di danni.
Oltre a questi ruoli tradizionali di mantenimento dell'inventario, i magazzini si sono evoluti per svolgere funzioni differenti (Maltz e DeHoratius, 2004):

- i centri cross-dock offrono una risposta rapida ed efficiente al consumatore poiché gli articoli sono consolidati con altre consegne pronte per la spedizione. L'obiettivo è la ricezione e la spedizione in giornata senza che la merce sia stoccata in magazzino. I prodotti tipici del cross-docking sono articoli deperibili che devono essere spostati rapidamente lungo la filiera;

- i centri di consolidamento ricevono prodotti da diverse fonti e creano dei pallet multiprodotto, personalizzati per la consegna ad un cliente o ad una linea di produzione che opera in *just-in-time*. Le scorte sono consolidate con quelle di altri fornitori per effettuare un'unica consegna al centro di distribuzione e l'obiettivo è quello di ottimizzare i costi e i trasporti, viaggiando *full-truck-load*. I centri di consolidamento differiscono dai centri cross-dock dal momento che il prodotto può rimanere nel magazzino per un periodo di attesa più lungo, aspettando la consegna alla destinazione finale;

- centri di servizi a valore aggiunto che svolgono attività varie, come ad esempio, la determinazione del prezzo e l'etichettatura delle merci per i clienti;

- magazzini per la logistica inversa che gestiscono imballaggi, resi o merci difettose;

- centri di assistenza e riparazione in cui i prodotti sono controllati, riconfezionati, riciclati o smaltiti.

I magazzini appena descritti possono essere di proprietà delle imprese oppure possono essere gestiti da fornitori di servizi logistici di terze parti (3PL) che svolgono operazioni dedicate per conto di un singolo cliente o per utenti condivisi. In questo modo, le aziende riescono a raggiungere economie di scala attraverso la condivisione di strutture, attrezzature e costi del lavoro.

2.2 L'automazione nei magazzini

I moderni sistemi di stoccaggio possono essere classificati come tradizionali o automatizzati. Nei sistemi convenzionali si utilizza una combinazione di manodopera e attrezzature di movimentazione per facilitare la ricezione, l'elaborazione e la spedizione dei prodotti. In generale, la manodopera costituisce una percentuale elevata dei costi complessivi. Pertanto, i sistemi automatizzati tentano di ridurre quanto più possibile il lavoro umano indirizzando gli investimenti di capitale in attrezzature innovative. Inoltre, ci si aspetta che un sistema automatizzato funzioni più velocemente e con maggiore precisione rispetto a un sistema tradizionale.

Il passaggio dalla struttura tradizionale a quella automatizzata è avvenuto grazie all'avvento dell'Industria 4.0. Molte organizzazioni hanno cambiato la loro strategia e hanno riflettuto

sull'utilizzo di tecnologie innovative per rendere le attività di magazzino più rapide e generare efficienza per ridurre i costi e l'intensità di lavoro.
I magazzini tradizionali erano considerati centri di costo e raramente come processi a valore aggiunto (Richards G., 2011): essi erano progettati e installati in modo tale da lavorare in uno spazio in cui alcuni fattori limitavano la superficie disponibile. Ulteriori aspetti negativi riguardavano un'elevata manipolazione dei materiali con conseguente spreco di tempo e denaro, un disequilibrio tra il tempo di attività e inattività delle risorse e, infine, una gestione non in grado di pianificare adeguatamente l'efficienza e le funzionalità delle operazioni.

La necessità di globalizzare il business, la crescita delle tecnologie e l'aumento della domanda degli utenti finali hanno visto cambiare la percezione del magazzino con la conseguente evoluzione al magazzino automatizzato. La struttura dello *smart warehousing* è simile a quella dei magazzini tradizionali in termini di layout e design, ma la differenza principale tra le due opzioni è il maggior fattore di sfruttamento superficiale e il tipo di tecnologie utilizzate.

Automatizzando tutte le operazioni possibili, si raggiunge un'elevata efficienza in termini di eliminazione delle operazioni ridondanti e riduzione del tempo di esecuzione delle attività di magazzino (Van den Berg, J. P., 2007).
Esistono diversi aspetti che possono essere automatizzati all'interno di un magazzino come, ad esempio, le operazioni di picking, la scansione di codici a barre e il trasporto tramite veicoli.

Soluzioni per il picking sono le tecnologie Goods-to-Person (GTP) che permettono di movimentare i prodotti verso il lavoratore, evitando che quest'ultimo si sposti verso l'oggetto che deve essere prelevato. I magazzini gestiti con sistema GTP ricevono ordini da un database centrale come un ERP, localizzano in modo automatizzato gli item e li movimentano nell'area dedicata al picking dove è presente un operatore. Tipicamente il *throughput rate* di un magazzino che usa tecnologia "merce verso l'uomo" è maggiore rispetto alla capacità effettiva di un magazzino tradizionale. Tali tecnologie permettono di avere costi operativi minori e un tasso di evasione degli ordini maggiore; inoltre, dal punto di vista del consumatore, il livello di servizio offerto risulta essere migliore.
Un'applicazione di questa tecnologia è l'Automated Storage and Retrieval System (AS/RS) che sarà presentata nel paragrafo successivo.

Un'altra soluzione alle operazioni di picking sono i sistemi Pick-to-light in cui dei LED guidano gli operatori alla corretta postazione di prelievo e indicano quanti prodotti devono essere prelevati. Ciò consente di migliorare l'accuratezza dell'inventario grazie alla riduzione dell'errore umano.
L'operatore, dopo aver svolto l'operazione di picking e depositati gli oggetti prelevati in un contenitore, conferma il completamento del task premendo un tasto vicino al display.

Infine, un metodo per automatizzare i trasporti riguarda l'implementazione di sistemi a guida autonoma detti Mobile Industrial Robot (MIR). Si tratta di robot in grado di ottimizzare le operazioni

logistiche e manifatturiere e capaci di interagire sia con gli operatori sia con i nastri trasportatori del magazzino per prelevare e depositare autonomamente i prodotti.

Una criticità rispetto ai magazzini tradizionali è, ad esempio, che un'anomalia del software potrebbe causare l'arresto dell'intera operazione poiché i sistemi sono altamente interconnessi. Inoltre, i guasti possono essere molto onerosi in termini di costi di riparazione e tempi di inattività.
Una barriera all'ingresso che impedisce a molte aziende la transizione ai sistemi automatizzati riguarda gli alti costi iniziali. Inoltre, dal punto di vista economico, l'elevato costo del capitale spesso richiede diversi anni per ottenere un ritorno economico o un ROI minimo.
Le imprese che riescono ad accedere a queste tecnologie potrebbero ridurre i costi delle operazioni fino al 30% e la perdita di scorte fino al 75%, e, inoltre, potrebbero migliorare le operazioni attraverso una maggiore agilità ed efficienza (Alicke K., 2017); un buon esempio, in questi anni, è stato Amazon.

2.3 Automated Storage and Retrieval System (AS/RS)

Gli Automated Storage and Retrieval System (AS/RS) sono ampiamente utilizzati nel settore della logistica (Allesina et al., 2010) in tutto il mondo sin dalla loro introduzione negli anni '60.
Si tratta di un magazzino che, sfruttando un dispositivo automatico, movimenta gli articoli all'interno della scaffalatura ed esegue le operazioni di input e output evitando lo spostamento dell'operatore *(Figura 2.1).*
I principali vantaggi rispetto ai sistemi di stoccaggio tradizionali sono l'elevato utilizzo dello spazio, la riduzione dei costi di manodopera, i brevi tempi di prelievo e il miglioramento del controllo delle scorte (Boysen e Stephan, 2016) e ciò consente una riduzione del consumo energetico stimolando la tendenza delle nuove supply chain green. Rispetto ai sistemi di archiviazione manuali, l'AS/RS può risparmiare fino al 70% del tempo di viaggio (Lee e Schaefer, 2007).

Gli AS/RS comportano un investimento significativo, pertanto la loro implementazione richiede un'analisi accurata durante la fase di progettazione iniziale (Roodbergen e Vis, 2009), poiché servirà a determinare le performance del sistema.
Le tipiche decisioni di progettazione relative agli AS/RS riguardano la scelta del sistema, cioè il tipo di macchina per la movimentazione, il numero di livelli, di corridoi e le dimensioni delle scaffalature, così come le regole di assegnazione dello stoccaggio e dei punti di sosta per i veicoli (Roodbergen e Vis, 2020). I componenti principali di un sistema AS/RS sono: scaffalature, vani di deposito, macchine di stoccaggio e prelievo, punti di ingresso e di uscita (I/O) e nastri trasportatori (Lerher et al., 2010).

Gli AS/RS consentono di migliorare l'efficienza operativa, in particolare in contesti caratterizzati da un'alta densità di componenti o materie prime di piccole e medie dimensioni (Lagorio et al., 2020).

Inoltre, sono anche denominati “intensivi”, con riferimento all’elevato grado di utilizzo dello spazio e possono funzionare 24 ore al giorno con una supervisione umana minima.

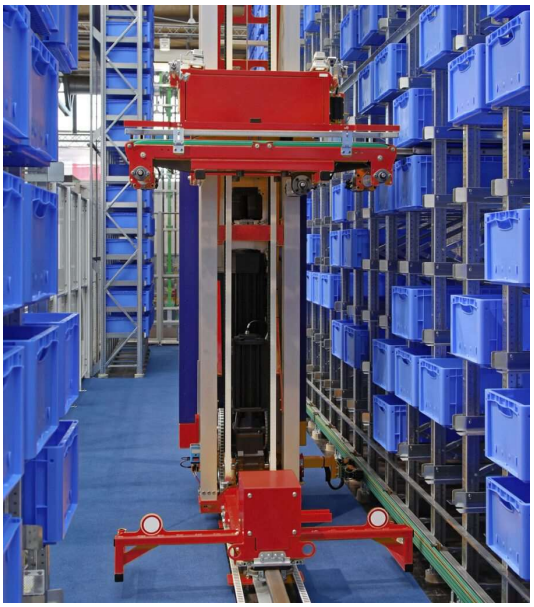


***Figura 2.1** – AS/RS (Fonte: berkshiregrey.com)*

2.3.1 Classificazione dei sistemi

La tecnologia AS/RS varia in modo sostanziale e può essere costituita da shuttle, crane, carousel, moduli di sollevamento verticale (VLM), mini-load o unit-load. Insieme al magazzino è spesso integrato un software di gestione, ovvero un Warehouse Management System (WMS) o altri controlli. Il contesto reale più diffuso è rappresentato da un trasloelevatore automatizzato che opera all’interno di una corsia utilizzando movimenti sia orizzontali che verticali (Boysen e Stephan, 2016) ed è in grado di posizionare e recuperare automaticamente i carichi da posizioni specifiche.

Le due categorie principali di magazzini AS/RS includono sistemi di carico unitario (unit-load) o sistemi di mini-carico (mini-load).

Gli AS/RS di tipo unit-load sono impiegati per carichi di grandi dimensioni e peso che superano i 500 Kg, come, ad esempio, la movimentazione di pallet. La macchina per la movimentazione utilizzata negli unit-load è una gru (crane) a corridoio fisso o a corridoio mobile. Le gru del primo tipo rimangono fisse su un’area o una fila di pallet e si spostano seguendo un percorso designato per recuperare gli oggetti *(Figura 2.2).*

Al contrario, le gru a corridoio mobile, sebbene abbiano un funzionamento analogo, sono progettate per recuperare o immagazzinare oggetti in più aree anziché lungo un percorso *(Figura 2.3)*.

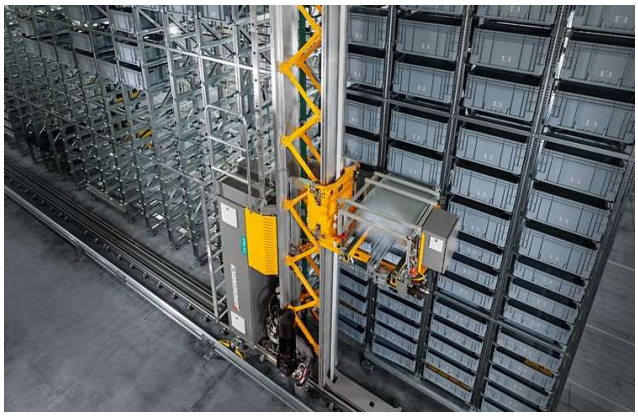


Figura 2.2 – *Gru a corridoio fisso*
(Fonte: jungheinrich.ch)

Figura 2.3 – *Gru a corridoio mobile*
(Fonte: jungheinrich.ch)

Gli AS/RS di tipo mini-load sono simili alla precedente tipologia ma movimentano carichi di minor peso e dimensione. In genere, i mini-load si servono di una gru (crane) o di navette (shuttle) per la movimentazione degli oggetti. Nel caso in cui il sistema si basa su un dispositivo a gru si ottiene una configurazione corrispettiva all'unit-load, tuttavia in versione assai ridotta, in cui si stoccano e recuperano prodotti lungo corridoi fissi e stretti *(Figura 2.4)*. Le navette dei mini-load, invece, si muovono su un binario e gestiscono oggetti tra sistemi di scaffalature automatizzate *(Figura 2.5)*.
Negli ultimi anni, in molte applicazioni industriali, si è visto aumentare l'utilizzo degli AS/RS che adottano gli shuttle per la movimentazione dei prodotti. L'ampio sfruttamento di questi macchinari è dovuto al fatto che le performance relative al *throughput*, alla flessibilità, alla scalabilità, alla riduzione del lavoro e all'efficienza energetica sono migliori rispetto a quanto è possibile ottenere in un magazzino gestito con altri sistemi di movimentazione. Il punto di forza di questo tipo di AS/RS è la possibilità di supportare elevate frequenze di ordini di picking, aspetto molto richiesto dal mercato.
La configurazione di questi magazzini è molto flessibile e adatta a soddisfare differenti necessità. Ad esempio, in contesti in cui è più rilevante la quantità di merce stoccata rispetto alla rapidità di esecuzione delle operazioni, è possibile progettare un magazzino con vani a doppia o tripla o quadrupla profondità. Logicamente, posizionando più oggetti all'interno della stessa locazione, si possono provocare dei rallentamenti in quanto, talvolta, è necessario movimentare dei prodotti prima di poter prelevare gli item richiesti in profondità. Per mitigare l'eccessivo aumento dei tempi relativi

alle movimentazioni interne, anche le navette possono essere dotate di due o più locazioni a bordo per il prelievo di più item contemporaneamente.

***Figura 2.4** – Crane mini-load*
(Fonte: viastore.com)

***Figura 2.5** – Shuttle mini-load*
(Fonte: conveyco.com)

Un'ulteriore tipologia di magazzini automatici a sviluppo orizzontale o verticale sono i caroselli *(Figura 2.6)*. La maggior parte degli AS/RS dotati di caroselli gestiscono oggetti più piccoli rispetto agli AS/RS di tipo unit-load.

Il funzionamento si basa su contenitori posizionati all'interno della scaffalatura che ruota orizzontalmente o verticalmente su un binario in acciaio e porta gli oggetti all'operatore situato nella zona di picking per il prelievo. Questa soluzione raggiunge performance considerevoli come maggiore velocità e precisione nell'evasione degli ordini e rapido accesso ai materiali immagazzinati con notevole riduzione dei tempi.

Infine, gli AS/RS basati su moduli di sollevamento verticale (VLM) funzionano in modo simile ai sistemi basati su carosello. In questo caso, tuttavia, un dispositivo automatico situato al centro di una scaffalatura chiusa individua il vassoio corretto fra quelli immagazzinati sui livelli verticali e lo consegna ad un *picker* che completa l'ordine e restituisce il vassoio. Con questa soluzione è possibile movimentare sia scatole di dimensioni notevoli sia oggetti di formato ridotto e, anche in questo caso, si riescono ad ottenere notevoli risparmi di spazio e di tempo.

***Figura 2.6** – Carosello orizzontale (Fonte: kardex-remstar.com)*

2.3.2 Variabili decisionali

L'efficienza di un AS/RS non dipende solo dalle caratteristiche descritte fino a questo punto ma anche dalle politiche di controllo come, ad esempio, la *dwell point policy* e l'*inventory policy*.
Per dwell point policy, si intende lo studio e la decisione della posizione in cui il trasloelevatore risiede quando è inattivo. Il punto di sosta è selezionato in modo tale da ridurre al minimo il tempo di viaggio previsto fino alla posizione della prima transazione dopo il periodo di inattività.

Le tre principali politiche dwell point prevalenti nelle soluzioni shuttle-based sono:

- *Stay dwell point:* la navetta resta nella posizione in cui conclude la missione, pertanto, la posizione occupata sarà l'output nel caso di missione di prelievo, viceversa, presso l'ultimo vano visitato;

- *Return to middle:* il trasloelevatore ritorna al centro dello scaffale ogni volta che è conclusa una missione;

- *Return to input:* al termine di ogni ciclo relativo ad una missione, il trasloelevatore torna al punto di ingresso;

- *Return to output:* al termine di ogni missione, lo shuttle torna al punto di uscita.

Con inventory policy si comprende un insieme di regole che determinano l'assegnazione delle locazioni alle unità di carico (u.d.c.) di diversi prodotti in un magazzino. Per quanto riguarda una prima distinzione tra le politiche di stoccaggio è possibile identificare: le *dedicated storage policy* e le *shared storage policy*.
Nelle politiche di archiviazione dedicata è richiesto di assegnare ad ogni vano del magazzino un'unica tipologia di prodotto per l'intero orizzonte di pianificazione.
Le politiche di stoccaggio condiviso, invece, consentono lo stoccaggio successivo di unità di prodotti diversi nella stessa ubicazione.
Il confronto tra la due politiche fa emergere che la shared storage policy garantisce il potenziale per ridurre la massima area di stoccaggio effettiva, per un uso più flessibile e una migliore accessibilità alle locazioni prioritarie rispetto alla dedicated storage policy. Questi fattori riducono il tempo medio di viaggio che, secondo Hausman W.H. nel Management Science, appartiene agli indici che sono generalmente utilizzati per valutare gli AS/RS, oltre all'utilizzo della macchina e alla capacità del magazzino in relazione al livello di servizio.

Tra le politiche di assegnazione dei vani, utilizzate sia per le dedicated storage sia per le shared storage policy, sono considerate, la random storage assignment, il turnover-based assignment, la nearest open position storage assignment e la class-based storage assignment (Hausman et al., 1976).

La random storage policy assegna la stessa probabilità a ciascun prodotto di essere immagazzinato in una delle qualsiasi posizioni disponibili; talvolta, questa politica viene utilizzata per approssimare la nearest open position storage assignment.
Il secondo tipo di politica è basata sul turnover ovvero il tasso di rotazione degli articoli. Esso rappresenta il numero di volte in cui si rinnova completamente un prodotto all'interno di un magazzino in un determinato arco di tempo e si calcola come il rapporto tra le vendite e la giacenza delle scorte. Seguendo questo criterio, i prodotti sono raggruppati in un numero ridotto di classi di un determinato valore di turnover e sono assegnati ad un ugual numero di zone in cui è suddiviso il magazzino. Tale assegnazione è effettuata in modo tale che i prodotti con maggior indice di rotazione siano più vicino alla zona di input e output del magazzino.

3. Simulation Modeling

La simulazione è uno strumento per creare modelli che imitano il comportamento di sistemi complessi con l'obiettivo di migliorarne la comprensione e identificarne le opportunità di miglioramento. La simulazione non serve ad ottimizzare poiché non si avvale di una funzione obbiettivo da massimizzare o minimizzare, bensì riproduce l'evoluzione del sistema nel tempo e analizza gli indici di performance in relazione a differenti scenari caratterizzati da specifiche condizioni di input.

I principali vantaggi riguardano, in primo luogo, il risparmio economico nel simulare diverse alternative in un ambiente virtuale piuttosto che realizzare onerosi cambiamenti nel sistema reale e di poter comprimere o espandere il tempo simulato. In secondo luogo, la simulazione consente di predire le prestazioni di progetti relativi a sistemi sperimentali e di classificarli analizzando differenti trade-off, oltre che essere uno strumento per la diagnosi delle criticità di un sistema. La simulazione può essere d'aiuto a comprendere lo sviluppo di un sistema, a prepararsi al cambiamento e a costruire il consenso per metterlo in atto.

Dall'altra parte, gli svantaggi non sono molti ma hanno una rilevanza tale da preferire, qualora possibili, strumenti alternativi alla simulazione. In particolare, la costruzione di un modello richiede capacità specifiche ed esperienza, i relativi tempi e costi di programmazione sono considerevoli seppur inferiori rispetto alla simulazione reale e i risultati che si possono ottenere non sono risposte esatte, bensì delle stime. Inoltre, l'analisi degli output deve interpretare correttamente i risultati poiché predizioni errate, basate su analisi statistiche sbagliate e un'impropria comprensione del comportamento del sistema, sono rischi sempre presenti.
La modellazione richiede astrazione e semplificazione, d'altronde se ogni aspetto del sistema dovesse essere riprodotto nei minimi dettagli, il costo del modello potrebbe avvicinarsi a quello del sistema che si vuole modellare, venendo meno uno dei vantaggi principali della modellazione. Pertanto, un modello è progettato per catturare alcuni aspetti del sistema che sono rilevanti al fine di acquisire conoscenze e approfondimenti sul comportamento del sistema (Morris, 1967). Il giusto livello di dettaglio dipende dal tipo di problema che si vuole analizzare. Una buona capacità di modellazione consiste nella costruzione di un modello che sia il meno dettagliato possibile ma in grado di produrre risposte adeguate alle domande che si sono poste.

Una volta costruito il modello, la sperimentazione si evolve attraverso la generazione di molte repliche del sistema in modo da ottenere risultati con una certa affidabilità statistica.
L'applicazione della simulazione spazia in differenti ambiti, riguardo, ad esempio, l'analisi costi-benefici e la stima dei KPI in sistemi di produzione o centri di stoccaggio, l'ottimizzazione dei tempi di risposta in un network di comunicazione, la conduzione di allenamenti simulati per il personale militare e la valutazione delle attività logistiche di una azienda. Inoltre, oggi la simulazione è utilizzata da diversi addetti alla tecnologia, come progettisti di impianti e project manager. In particolare, le attività legate alla produzione e alle attività di reingegnerizzazione dei processi

aziendali utilizzano la simulazione per selezionare i parametri di progettazione, pianificare il layout della fabbrica, gli acquisti di attrezzature e perfino valutare i costi finanziari ed il ritorno sull'investimento.

3.1 Sistemi e modelli

Come illustrato nella *Figura 3.1 –,* dato un sistema, è possibile analizzarlo o sperimentando il sistema reale stesso, ad esempio con l'esecuzione di un crash test, o, in alternativa, attraverso la costruzione di un modello del sistema reale. A sua volta, un modello può appartenere a due diverse categorie: la prima riguarda il modello fisico, ovvero un oggetto fisico semplificato o ridimensionato che descrive i fenomeni reali, come, ad esempio, un modellino in scala di un centro di stoccaggio, mentre la seconda tratta il modello matematico. Quest'ultimo si suddivide in modello analitico, in cui si costruisce un modello di programmazione lineare che fornisce una soluzione numerica, e in modello di simulazione in cui non è possibile tradurre in equazioni lineari i fenomeni rappresentati; tuttavia, è caratterizzato da un modello software che imita il comportamento del sistema reale. Una soluzione ibrida, tra le due precedentemente esposte, è quella del meta-modello, ovvero un modello che comprende sia parti analitiche che parti risolvibili tramite la simulazione. Solitamente i meta-modelli si utilizzano per rappresentare dei sistemi molto complessi.

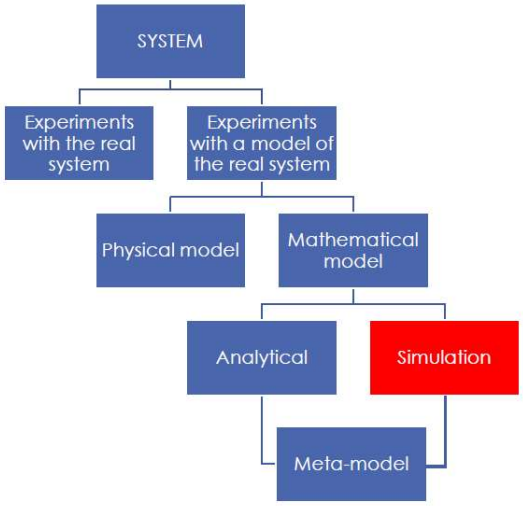


***Figura 3.1** – Analisi di un sistema*
(Fonte: Simulation Modeling and Analysis with Arena, 2007)

3.1.1 Confronto tra modello analitico e modello di simulazione

Il trade-off tra modellazione analitica e simulazione risiede nella natura delle soluzioni, cioè nel calcolo delle loro misure di performance:

- un modello analitico richiede la soluzione di un problema matematico che, successivamente, è utilizzata per ottenere le misure di performance interessate;
- un modello di simulazione richiede l'esecuzione di un programma per produrre uno storico di campioni da cui sono calcolate una serie di statistiche, ovvero delle stime delle misure di performance.

Un modello analitico è preferibile ad uno di simulazione quando è possibile determinare una soluzione esatta dal momento che il suo calcolo è di solito più veloce della simulazione. Tuttavia, i sistemi sono molto complessi e difficilmente possono essere modellizzati con un metodo analitico, preferendo quindi la simulazione. Inoltre, il modello analitico non è sempre in grado di essere sufficientemente potente da catturare gli aspetti comportamentali chiave del sistema. Contrariamente, la simulazione può modellizzare un sistema soggetto a qualsiasi insieme di ipotesi, anche se, a volte, può richiedere un grande sforzo umano in termini di progettazione del modello e di costo computazionale.

Per concludere, la versatilità dei modelli di simulazione e la fattibilità delle loro soluzioni superano di gran lunga quelle dei modelli analitici. Nella pratica, la simulazione è molto usata dagli ingegneri in un'ampia gamma di aree di applicazione grazie alla sua capacità di ricreare un ambiente di laboratorio virtuale, in cui sistemi progettati in modo diverso possono essere confrontati e valutati in sicurezza.

3.2 I modelli di simulazione

I modelli di simulazione si possono classificare secondo differenti aspetti, come la presenza o meno di variabilità in un sistema o la dipendenza dal tempo. In particolare, secondo la *Figura 3.2* – Classificazione modelli di simulazione, una prima distinzione identifica i modelli deterministici e i modelli stocastici. I primi sono caratterizzati dall'assenza di variabilità, perciò, a parità di dati in input si ottiene sempre la stessa risposta in output; per contro, i modelli stocastici possiedono variabilità sia in input sia in output del sistema e, dunque, necessitano di molte prove di simulazione per compiere valutazioni adeguate.

Una seconda distinzione riguarda le categorie di modelli statici o dinamici. Nel caso di un sistema reale che ha una evoluzione nel tempo allora si implementa un modello dinamico, dipendente dal tempo. Nel caso in cui il sistema sia osservato in un istante di tempo preciso o che sia indipendente dal tempo si parla di modello statico.

Infine, i modelli dinamici sono suddivisi in modelli discreti o modelli continui. I primi sono più semplici poiché il sistema si evolve in precisi istanti di tempo caratterizzati dal cambiamento delle variabili di stato; al contrario, i modelli continui sono più complessi in quanto le variabili di stato possono cambiare continuamente nel tempo. A livello pratico, nessun modello è completamente discreto o continuo, bensì si utilizzano modelli ibridi.

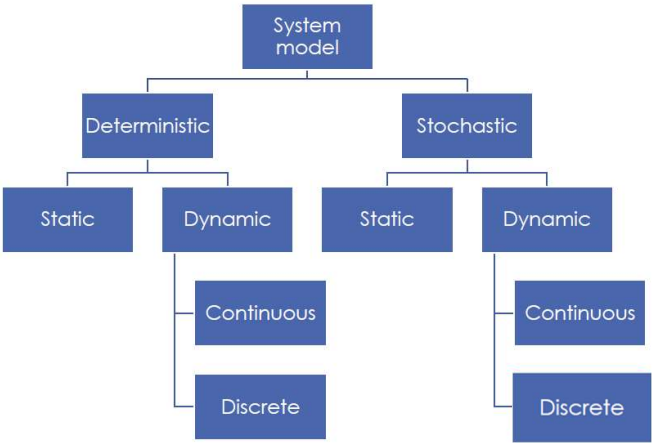


Figura 3.2 – *Classificazione modelli di simulazione*
(Fonte: Simulation Modeling and Analysis with Arena, 2007)

3.2.1 Discrete Event Simulation

La maggior parte degli strumenti di simulazione adotta il cosiddetto paradigma della simulazione ad aventi discreti (Banks et al.,1999), esso è utile per rappresentare sistemi grandi e complessi (Law e Kelton, 2000). Un sistema ad eventi discreti è un sistema dinamico i cui stati possono assumere valori logici o simbolici, piuttosto che numerici, e il cui comportamento è caratterizzato dall'occorrenza di eventi che si verificano in punti temporali discreti. Pertanto, il comportamento di tali sistemi è descritto, in termini di stati e di eventi.
Uno stato è definito vettoriale poiché rappresenta, in modo compatto, un insieme di dati, cattura le principali variabili del sistema e permette di descrivere la sua evoluzione nel tempo. Gli eventi individuano gli accadimenti che possono far cambiare lo stato del sistema. Essi sono contenuti in una lista ordinata cronologicamente per tener traccia dell'evento imminente e dei successivi. La simulazione si evolve nel tempo procedendo da un evento all'altro della lista.

In pratica, ogni volta che si verifica un evento, si aggiorna lo stato della simulazione, le statistiche appropriate e infine la lista. L'esecuzione di un evento può cambiare le variabili di stato e, eventualmente, schedularne altri che saranno aggiunti alla lista. Quindi si procede al prossimo evento elencato e l'algoritmo termina quando la lista è vuota.
Una caratteristica essenziale del paradigma DES è che nulla cambia lo stato, a meno che non si verifichi un evento. Tra un evento e l'altro, lo stato del modello di simulazione è considerato costante,

anche se il sistema è impegnato in alcuni processi.
Esempi di sistemi che sono valutati utilizzando la simulazione DES includono:

- Qualsiasi sistema di accodamento, come uno sportello di servizio bancario, dove i clienti arrivano occasionalmente, aspettano in fila per il servizio, ricevono il servizio e partono;
- Sistemi di produzione, in cui le parti vengono lavorate in varie sequenze in diverse stazioni, dopodiché lasciano lo stabilimento;
- Sistemi di inventario, in cui quantità casuali di un determinato prodotto vengono acquistate dai clienti ogni giorno in un negozio e in cui le forniture del prodotto si spostano da una fase all'altra della catena di approvvigionamento prima di essere acquistate presso il negozio.

Gli esempi precedenti sono guidati da diversi eventi che si verificano in momenti discreti e cambiano lo stato della simulazione, essi possono corrispondere agli arrivi o alle partenze dei clienti, ai guasti delle macchine o qualsiasi altro evento caratteristico di un sistema.

I modelli di simulazione di eventi discreti sono molto utili quando i componenti dei sistemi reali cambiano in seguito ad eventi specifici (Price, 2014). Dal momento che esistono molti sistemi reali caratterizzati dall'occorrenza di eventi in momenti discreti l'implementazione del paradigma DES alle simulazioni risulta compatibile ed efficiente.

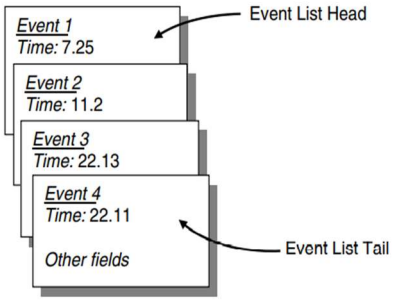


Figura 3.3 - *Struttura di una lista di eventi discreti*
(Fonte: Simulation Modeling and Analysis with Arena, 2007)

3.2.2 Agent-Based Modeling and Simulation

Agent-Based Modelling and Simulation (ABMS) è un approccio a supporto dei modelli di decision-making applicati ai sistemi. Consiste nella modellazione mediante agenti capaci di autogovernarsi con regole proprie e interagire a vicenda o con l'ambiente circostante (C. M. Macal e M. J. North,

2005). Pertanto, l'ABMS è definito come la simulazione di un sistema che include l'ambiente, le risorse e gli agenti che si comportano in maniera autonoma e indipendente (S. M. Sanchez e T. W. Lucas, 2002).

Nella forma base, l'ABMS fonde la simulazione a eventi discreti (DES) che fornisce un *framework* per la modellazione delle interazioni degli elementi del sistema, con la programmazione orientata agli oggetti che fornisce un *framework* per organizzare e gestire gli agenti in base ai loro comportamenti. Inoltre, l'uso di agenti intelligenti all'interno dell'ABMS consente di catturare i meccanismi che governano il sistema modellato (D. Chen et al., 2008).

Per quanto riguarda gli agenti, essi sono entità software capaci di percepire l'ambiente circostante, comunicare fra loro, modificare il proprio comportamento in base alle informazioni acquisite e, infine, eseguire autonomamente dei compiti assegnati per raggiungere un dato obbiettivo. La potenzialità di questo approccio risiede nella semplicità delle interazioni fra agenti che, tuttavia, viste nel loro insieme, consentono di descrivere con molta precisione i meccanismi fondamentali di un sistema reale e complesso.

L'ambiente è il mondo virtuale nel quale gli agenti operano e può essere sia un elemento neutro, nel senso che non ha nessuno effetto sugli agenti, o può essere realizzato con la stessa cura e complessità degli agenti stessi. Un vantaggio è che ABM è adattabile con facilità a cambiamenti del contesto. È da considerare che l'ambiente stesso può cambiare il comportamento del sistema e, automaticamente, può influenzare gli altri agenti. La possibilità di modellizzare tali interazioni agente-agente è la differenza principale rispetto ad altri tipi di modelli di simulazione.

Sin dagli inizi della sua introduzione, l'approccio ABM è stato riconosciuto come uno dei paradigmi più promettenti per svolgere investigazioni dettagliate e affidabili dei problemi relativi ai complessi sistemi del mondo reale. Un esempio di applicazione della modellazione ad agenti riguarda le supply chain, esse, infatti, sono intrinsecamente adatte a tale descrizione in quanto si tratta di una rete distribuita e collaborativa di attori aziendali, interconnessi da flussi fisici, flussi informatici, flussi finanziari ed altro ancora (M. S. Fox et al., 2000). Inoltre, la modellazione ad agenti rappresenta uno strumento per la pianificazione e gestione dei rischi delle supply chain (F. Nilsson e V. Darley, 2006).

I principali vantaggi dell'approccio agent-based sono la grande flessibilità che consente di cambiare le regole esistenti con nuove condizioni per individuare le migliori performance del sistema e la modularità, ovvero la possibilità di suddividere lo sviluppo della programmazione in moduli che svolgono specifiche funzioni (C.S. Taber e R.J. Timpone, 1996).

L'agent-based modelling ha la capacità di collezionare dati autonomamente e supportare il processo di decisione basandosi sull'analisi sia delle serie storiche sia delle previsioni future. Si utilizza, ad esempio, per definire quali materiali stoccare sullo stesso scaffale al fine di ottimizzare le operazioni di picking, come gestire le spedizioni dei *truck* negli orari di minimo traffico o come individuare il percorso ottimo per spedire i prodotti.

In aggiunta, l'approccio basato sugli agenti è più simile alla realtà rispetto ad altre tipologie di modellazione poiché gli agenti sono in grado di incorporare i meccanismi di simulazione ad eventi discreti e, di conseguenza, possono modellizzare la variabilità e gli eventi intrinseci del sistema.

Infine, forniscono un ambiente di simulazione e analisi molto completo per confrontare soluzioni alternative utilizzando differenti condizioni basate sulle misure di prestazione. Gli ABMS possono anche essere ibridati con metodi come il DOE, in quanto sono in grado di modellare eventi come ordini, spedizioni, guasti alle macchine e così via, in modo molto dettagliato (J. P. C. Kleijnen, 2005).

3.3 La simulazione applicata ai magazzini

Il gran numero di variabili che influenzano le prestazioni e i costi dei magazzini rende difficile prendere decisioni efficaci durante la fase di ottimizzazione del magazzino. Pertanto, strumenti e metodi sono essenziali per supportare i processi decisionali e gestionali. In particolare, l'uso di simulazioni è stato studiato da diversi ricercatori (Bonini, 1963; Macro e Salmi, 2002; Lee et al, 2003; Chen et al, 2013) come approccio alternativo alle tecniche convenzionali. In molti campi pratici, la simulazione è il metodo più accessibile per capire come le numerose variabili interagiscono e condizionano le prestazioni dei sistemi.

Tuttavia, nel campo della logistica di magazzino la simulazione è ancora a uno stadio primitivo: da una parte, si nota come la simulazione è stata ampiamente applicata nell'ambito delle supply chain, mentre, dall'altra sono stati concentrati molti meno sforzi sulle operazioni di magazzino o dei centri di distribuzione nonostante la simulazione, sviluppata in questa direzione, presenta un potenziale promettente per le applicazioni future.

In ambito logistico, gli obiettivi che la simulazione si pone di raggiungere sono:

- aiutare i manager a valutare le prestazioni delle diverse strategie di stoccaggio e prelievo. In particolare, si ritiene che l'ottimizzazione dell'assegnazione delle posizioni di stoccaggio ai prodotti possa contribuire a efficientare le operazioni;
- individuare la capacità ottima del magazzino in relazione alla crescita dell'azienda;
- valutare i colli di bottiglia e i vincoli del sistema;
- migliorare il flusso dei prodotti;
- ottimizzare i tempi relativi alle performance di magazzino.

Gli studi di simulazione non solo aiutano a comprendere i dettagli dei processi, ma gli strumenti di modellazione grafica e la rappresentazione dinamica degli oggetti, facilitano il coinvolgimento del management nello sviluppo e nei processi decisionali (Seila et al., 2003).

Una metodologia di simulazione relativa ai magazzini e basata sugli agenti, descritta da Ribino P. nel 2015, combina tre prospettive diverse sulla modellazione del magazzino: il modello ambientale, il modello organizzativo e il modello comportamentale.

Il primo rappresenta le caratteristiche fisiche e funzionali di un magazzino e comprende gli attori che lo popolano, gli oggetti che ne determinano gli aspetti fisici, l'insieme delle azioni che gli attori possono eseguire per completare alcuni processi e, infine, i comportamenti che ogni attore è in grado di gestire, ovvero, i concetti specifici che appartengono al suo dominio.

Il secondo si occupa di definire lo schema organizzativo adottato dalle entità coinvolte nei processi di magazzino.

Il terzo descrive il comportamento dinamico di ogni ruolo nel modello organizzativo e si basa su due elementi principali: i piani che includono un insieme di azioni per definire il comportamento di un'entità nel magazzino e gli eventi che potrebbero potenzialmente innescare cambiamenti sul sistema.

I modelli agent-based sono considerati più generici e più potenti in confronto ad altri approcci di modellazione in quanto sono in grado di descrivere sistemi più complessi e dinamici (Borshchev and Filippov, 2004).

In conclusione, sono presentate alcune applicazioni della letteratura. Ad esempio, Burinskiene et al. nel 2018, hanno presentato un modello di simulazione di vari processi di trasporto all'interno di un magazzino nel tentativo di ridurre gli sprechi. I risultati hanno indicato che il 67% degli sprechi può essere evitato. Merschformann et al. nel 2018 hanno simulato i processi di prelievo e di rifornimento, l'assegnazione degli ordini e i problemi di selezione e di assegnazione delle locazioni in un sistema robotizzato di movimentazione dei materiali valutando più regole decisionali per ogni problema.

Infine, i lavori di Dohrmann et al. nel 2019 applicati all'azienda DHL hanno fatto luce sull'effetto dei Digital Twin di magazzino sull'ottimizzazione di una struttura logistica. In particolare, utilizzando un modello 3D alimentato da dati IoT, i DT consentono alle aziende di creare scenari di simulazione affidabili, prevedendo il movimento dei prodotti, del personale e delle attrezzature di movimentazione dei materiali, contribuendo così all'ottimizzazione dell'utilizzo dello spazio, alla riduzione della spesa energetica e al miglioramento della produttività dei dipendenti. DHL ha implementato questa tecnologia per il magazzino dell'azienda Tetra Pak, un gruppo internazionale di packaging (Rusch B., 2019).

4. Metodologia

Il progetto proposto dal Dipartimento di Ingegneria Gestionale e della Produzione (DIGEP) del Politecnico di Torino prevede la realizzazione di un Digital Shadow del magazzino automatico presente nel Laboratorio di Eccellenza. L'obiettivo dello studio è la creazione di un modello di simulazione attraverso il software Anylogic e la successiva ottimizzazione delle operazioni svolte dal trasloelevatore.

Seguendo gli step principali della modellazione di un sistema di simulazione, la ricerca si compone delle seguenti fasi:

1. Analisi del sistema reale;
2. Costruzione del modello concettuale;
3. Creazione del modello virtuale;
4. Sviluppo di un flow chart di ottimizzazione e del relativo pseudocodice;
5. Analisi dei risultati.

In questo capitolo sono presentate le metodologie utilizzate per sviluppare gli step del progetto, successivamente, nel capitolo 5, viene descritto il sistema reale. Nel capitolo 6, è descritto il modello concettuale applicato (UML), mentre nel capitolo 7 si presenta il sistema virtuale realizzato. Lo sviluppo di un flow chart di ottimizzazione e del relativo pseudocodice è analizzato nel capitolo 8 e, infine, nel capitolo 9, vengono proposti i risultati della ricerca.

4.1 Analisi della costruzione di un modello

Per ottenere una modellazione di qualità bisogna seguire i seguenti step fondamentali.

1- Analisi del problema e raccolta dati

Il primo passo nella costruzione di un modello di simulazione consiste nell'analizzare il problema stesso. Pertanto, si identificano i parametri di input, i KPI, le relazioni tra parametri e variabili e i meccanismi di funzionamento dei componenti del sistema. In questa fase, si effettuano delle semplificazioni e delle assunzioni per far fronte ad alcuni aspetti troppo complessi da essere analizzati.

Per quel che riguarda la raccolta dei dati, essa è necessaria per stimare i parametri di input del modello. Quando non ci sono i dati, potrebbe essere ancora possibile designare intervalli di parametri e simulare il modello per tutti o alcuni valori di input. La raccolta dei dati è necessaria anche nella fase di convalida del modello, cioè, i dati raccolti sulle statistiche di output del sistema reale sono confrontati con le loro controparti del modello.

2- Modello concettuale

La modellazione concettuale è una fase preliminare alla costruzione del modello software in cui si rappresenta il sistema attraverso strumenti di visualizzazione. Si tratta di un processo non lineare bensì ciclico in cui il progettista e il committente si confrontano per valutare quali aspetti includere o meno nella realizzazione del software. Il modello concettuale deve contenere tutte le assunzioni e le semplificazioni concordate nella fase di analisi del problema.
Gli strumenti di visualizzazione più utilizzati sono, ad esempio, il flow chart che ha un formalismo grafico e rappresenta il flusso logico del sistema, l'Unifed Modeling Language (UML) che è un linguaggio di modellazione e di specifica basato sul paradigma object-oriented, o ancora, l'ERG che è il grafo degli eventi in cui i nodi rappresentano gli eventi e gli archi le relazioni tra gli eventi.

Nel diagramma UML, ogni classe, rappresentata da un rettangolo, identifica un'astrazione di entità con caratteristiche comuni e ogni collegamento identifica una relazione statica tra le classi. Una classe è costituita da un nome, da una serie di attributi (parametri e variabili) e dalle funzioni che svolge. Possono essere presenti relazioni semplici che definiscono il ruolo di un'entità rispetto ad un'altra, oppure, associazioni di generalizzazione, una relazione in cui un elemento del modello (il secondario) dipende da un altro elemento del modello (il principale).
Ogni relazione è inoltre corredata dalle cardinalità. Quest'ultima indica il numero di istanze di una classe che possono essere associate ad una singola istanza dell'altra classe.

In ultimo, lo strumento è consegnato ad uno sviluppatore per la creazione del software.

3- Costruzione del modello software

La principale decisione da effettuare nella costruzione del modello software riguarda il linguaggio informatico impiegato. Esistono due macrocategorie di linguaggi, da una parte, i linguaggi general-purpose, come c++, java, visual basic e Python, dall'altra parte, i linguaggi specific-purpose, come Anylogic e Arena. I primi, permettono di programmare a livello di singola istruzione consentendo il massimo della flessibilità, mentre, i secondi, sono caratterizzati da costrutti predefiniti.

4- Ottimizzazione del modello

Dopo aver sviluppato il modello base su Anylogic, la fase successiva è l'introduzione di un processo di ottimizzazione nel modello digitale. Nel sistema reale, il trasloelevatore implementa un algoritmo di ottimizzazione, il cosìdetto algortimo di Dijkstra, per gestire i task presenti in una sessione affinchè il tempo di esecuzione sia il minore possibile.

L'algoritmo di Dijkstra,ideato nel 1956 dall'informatico Edsger W. Dijkstra da cui prende il nome, è un algoritmo usato per trovare il percorso più breve tra i nodi appartenenti ad un grafo, il quale

rappresenta, ad esempio, una rete stradale in cui i nodi sono le città. L'algoritmo esiste in molte varianti. L'algoritmo originale di Dijkstra trova il percorso più breve tra due nodi generici (Dijkstra, E. W. (1959)) , tuttavia, esiste una variante più comune che, fissato un singolo nodo denominato *source node*, trova il percorso più breve per raggiungere tutti gli altri nodi della rete. La soluzione trovata prende il nome di *shortest-path tree*. (Mehlhorn, K., & Sanders, P. (2008)).

Il processo di ottimizzazione può essere anche usato per trovare il percorso più breve da un singolo nodo di origine verso un singolo nodo di destinazione fermando l'algoritmo una volta che il percorso più breve è stato trovato.

L'algoritmo di Dijkstra risolve il problema del percorso più breve per qualsiasi grafo orientato ponderato con pesi non negativi. Può gestire grafi costituiti da cicli, ma i pesi negativi faranno sì che questo algoritmo produca risultati errati. Di conseguenza, assumiamo che w(e)>=0 per ogni $e \in E$.

Un grafo diretto G(N,L) è definito da un set N di nodi o vertici e un set L di collegamenti o archi. Questi ultimi sono rappresentati da coppie ordinate di nodi, pertanto, il collegamento dal nodo i-esimo al nodo j-esimo è diverso dal collegamento dal nodo j-esimo al nodo i-esimo.
Al contrario, possono esistere dei grafi indiretti o semplici, caratterizzati da una relazione reciproca tra i vertici. In altre parole,due differenti nodi di un grafo sono collegati da un unico arco percorribile in entrambe le direzioni.
Un grafo ponderato è un grafo in cui a ciascun arco è assegnato un peso numerico. Tale peso può rappresentare il costo o la distanza necessaria per viaggiare da un nodo all'altro, oppure la capacità, cioè la quantità di flusso massima che può essere trasportata da un nodo all'altro. Un grafico ponderato è un tipo speciale di grafico etichettato in cui le labels sono numeri, solitamente considerati positivi .

5- Verifica del modello

Lo scopo della verifica è assicurarsi che il modello sia costruito correttamente e sia conforme alle sue specifiche e che il software funzioni come il progettista abbia pensato di crearlo. Esistono due approcci fondamentali alla fase di verifica: sperimentale e analitico. Il primo, consiste nello sperimentare il comportamento di un software per analizzare se questo si comporta secondo le aspettative. L'altro, consiste nell'analizzare il prodotto e tutta la documentazione di progetto ad esso relativa per ricavare indicazioni circa la correttezza delle operazioni svolte rispetto alle decisioni progettuali preliminari.

6- Convalida del modello

Definiti gli obbiettivi di un modello di simulazione essi diventano la leva su cui si valuta la validazione. Solitamente si dimostra la validazione o meno di un sistema sotto determinate condizioni, infatti, risulta troppo dispendioso convalidare un modello soggetto a qualsiasi ipotesi.
La fase di validazione non è completamente separata dalla fase di costruzione di un modello; si tratta di un processo iterativo in cui man mano che si avanza con la modellazione e la validazione, aumenta l'affidabilità. La convalida esamina l'adattamento del modello ai dati empirici raccolti sul sistema reale da modellizzare. Un buon fit del modello significa che un insieme di misure di performance previste dalla simulazione corrispondono ragionevolmente alle loro controparti osservate nel sistema reale. Naturalmente, questo tipo di convalida è possibile solo se il sistema fisico esiste e le misurazioni richieste possono effettivamente essere acquisite. Eventuali discrepanze significative suggerirebbero che il modello proposto è inadeguato ai fini del progetto e sarebbero necessarie modifiche.
L'analista solitamente impiega più tecniche di validazione contestualmente, inoltre, non esistono algoritmi che indicano quali strumenti possono risultare maggiormente efficaci in determinati casi anche se principalmente si fa uso dei test statistici.
La convalida del modello si conclude con la scrittura di una documentazione dei risultati ottenuti nonché delle metodologie utilizzate.

7- Analisi dell'output

Le misure di prestazione stimate sono oggetto di un'analisi statistica approfondita. Un problema tipico è quello di identificare i migliori design tra una serie di alternative. Un'analisi statistica esegue test di inferenza per comporre una classificazione dei progetti alternativi caratterizzati da misure di prestazione superiori.

8- Raccomandazioni finali al cliente finale

Per concludere, l'analista utilizza l'analisi degli output per formulare le raccomandazioni finali al committente. Generalmente, esse sono documentate in una relazione scritta che ha lo scopo di rispondere, attraverso i risultati della simulazione, al problema richiesto.

5. Descrizione sistema reale

Come emerso dal paragrafo relativo alla metodologia, il primo step da eseguire per la costruzione di un modello funzionante e validato è la descrizione del sistema reale.

Il magazzino automatico AS/RS, fornito dall'azienda Incas-SSI Schafer Group, è stato installato nel Laboratorio di Eccellenza del Politecnico di Torino ed occupa un'area rettangolare di dimensione pari a 7.60 x 3.40 metri *(Figura 5.1)*. Inoltre, è presente un corridoio ampio circa 2 m impiegato per le *operations* e le movimentazioni dei materiali. Il magazzino automatico consente sia di immagazzinare differenti tipologie di cassette di plastica sia di movimentarle all'interno della scaffalatura. In particolare, la struttura presente nel laboratorio è caratterizzata da una composizione che include, da una parte, una stazione per il picking manuale e, dall'altra, una per la preparazione di kit; entrambe dotate di flow rack a gravità con implementazione della tecnologia *pick-to-light* per il supporto delle attività operative. La prima postazione consente di preparare gli ordini destinati alla produzione o alla spedizione, la seconda, invece, permette di comporre dei kit con differenti item, destinati all'assemblaggio. A tal riguardo, è necessario che siano presenti in stock delle cassette vuote per garantire la possibilità di effettuare le operazioni e la movimentazione.

Infine, la scaffalatura comprende due *conveyor*, uno relativo alla movimentazione delle unità di carico in ingresso e uno all'uscita; inoltre, sono presenti anche due AGV capaci di interfacciarsi con le rulliere e che svolgono le movimentazioni all'esterno della scaffalatura; tuttavia, non sono stati considerati nella trattazione di questo lavoro di tesi.

***Figura 5.1** – Magazzino AS/RS presente nel laboratorio del DIGEP*

Nel dettaglio, le differenti aree del magazzino automatico sono suddivise come mostrato nella *Figura 5.2*:

1. Scaffalatura per le cassette;
2. Trasloelevatore Maxi Shuttle;
3. Flow rack a gravità per l'area di picking;
4. Flow rack a gravità per l'area kitting;
5. Area intermedia tra postazione di picking-postazione di spedizione;
6. Rulliera in ingresso dal magazzino;
7. Rulliera in uscita dal magazzino.

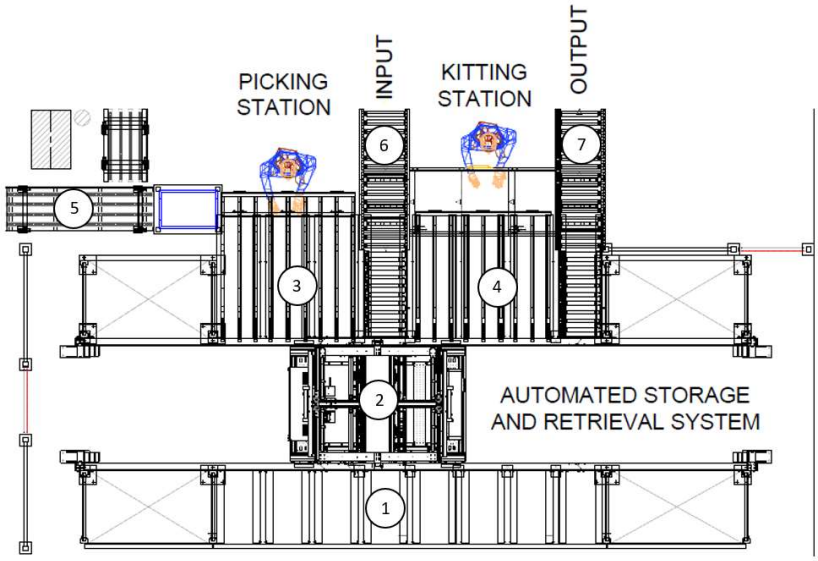


***Figura 5.2** – Aree magazzino DIGEP*

Di seguito le caratteristiche tecniche del magazzino automatizzato:

- Un sistema Maxi-Shuttle (MS) di tipo mini-load a corsia vincolata, simile a un mini trasloelevatore in grado di spostare cassette lungo tre assi ed equipaggiato di una forca per l'accesso alle diverse profondità di stoccaggio e per il prelievo dell'unità di carico *(Figura 5.3)*.
  Di seguito sono elencate alcune caratteristiche tecniche del Maxi-Shuttle:
  - Asse x: velocità max (4.0 m/s) accelerazione/decelerazione (1.5 m/s^2);
  - Asse y: velocità max (0.8 m/s) accelerazione/decelerazione (1.6 m/s^2);
  - Asse z: velocità max (0.5 m/s) accelerazione/decelerazione (1.5 m/s^2);
  - Dimensione della culla: 1200 x 400 mm;

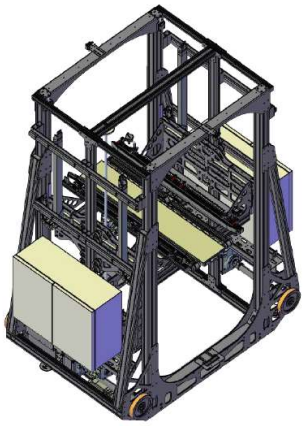


***Figura 5.3** – Trasloelevatore Maxi-Shuttle DIGEP (Fonte: Incas SSI Schafer, 2020)*

- Il numero dei vani totali ammonta a 101, suddivisi in 90 dedicati allo stoccaggio, 1 all'input e 10 all'output. I vani di stoccaggio sono progettati di differenti dimensioni, una metà di altezza 225 mm e l'altra di 338 mm, per ospitare unità di carico di diverse tipologie. Un ulteriore aspetto da considerare è che 22 dei vani di stoccaggio sono in quadrupla profondità e 68 in doppia. In ultimo, ciascun vano può immagazzinare contemporaneamente solo un tipo specifico di cassette;

- I due fronti della scaffalatura a corsia singola sono costituiti da 8 colonne e 7 livelli ciascuno: il fronte 1 ospita i flow rack del picking e del kitting oltre che una parte dei vani di stoccaggio, i restanti dei quali sono ripartiti, in numero superiore, sul fronte 2;

- Le cassette sono di cinque tipologie differenti *(Tabella 5.1)*. In particolare, le unità di carico sono basse (120 mm) o alte (220 mm) e si collocano in vani compatibili alle proprie dimensioni; inoltre, possono essere corte (300 mm) o lunghe (600 mm) e occupare, rispettivamente, una o due posizioni nei vani *(Figura 5.4, Figura 5.5, Figura 5.6, Figura 5.7)*. Infine, esiste il vassoio, un'unità di carico che ha dimensioni uniche ed è considerato flessibile in quanto può essere immagazzinato assieme a cassette differenti *(Figura 5.8)*. Per il tracciamento all'interno del magazzino è stato assegnato un codice univoco a ciascuna unità di carico;

Tabella 5.1 *– Tipologie cassette*

TIPO	DESCRIZIONE	CODICE	LUNGHEZZA (mm)	LARGHEZZA (mm)	ALTEZZA (mm)	POSIZIONI OCCUPATE
LTB 6120	Cassetta grande bassa	1001-1030	600	400	120	2
LTB 6220	Cassetta grande alta	2001-2050	600	400	220	2
LTB 4120	Cassetta piccola bassa	3001-3020	300	400	120	1
LTB 4220	Cassetta piccola alta	4001-4020	300	400	220	1
MF 6070	Vassoio	5001-5010	600	400	70	2

Figura 5.4 *– Cassetta LTB6120*

Figura 5.5 *- Cassetta LTB6220*

Figura 5.6 - *Cassetta LTB4120*

Figura 5.7 - *Cassetta LTB4220*

Figura 5.8 - *Cassetta MF6070*

- Le rulliere sono motorizzate e ricoprono principalmente il ruolo di movimentazione in ingresso e in uscita dal magazzino ***(Error! Reference source not found.)***. Il movimento delle unità di carico è sul lato corto. Il movimento delle unità di carico è sul lato corto.
  Di seguito sono elencate alcune caratteristiche tecniche dei trasportatori installati:
  -Dimensioni del piano di carico: 800 x 600 mm;

- Area di trasporto: 890 x 485 mm;
- Carico utile massimo: 25 kg;
- Regolazione manuale o automatica dell'altezza da 650 mm a 850 mm;
- Tempo di carico/scarico: 3 s.
- Velocità 0.5 m/s;
- Accelerazione/decelerazione 0m/s.

***Figura 5.9** - Rulliera*

- Per la sicurezza degli operatori è installata una rete metallica per delimitare l'area di lavoro del trasloelevatore e, nelle parti esposte, sono montati pannelli di protezione in plexiglas.

Il sistema reale include anche un sistema informativo e tecnologico per la gestione e la movimentazione dei materiali. La gerarchia del flusso informativo si sviluppa su quattro livelli differenti che individuano aspetti via via più operativi *(Figura 5.10)*.

Il quarto livello, relativo all'ERP, non è ancora stato implementato nel laboratorio e sarà oggetto di sviluppi futuri. L'acronimo ERP si riferisce all'Enterprise Resource Planning, un software per pianificare e gestire tutti i processi principali di un'organizzazione come, ad esempio, la pianificazione delle richieste di produzione rispetto alle relative distinte base.
Ogni qual volta che è avviato un piano di produzione, l'elenco delle distinte base coinvolte è inviato al WMS. In particolare, il WMS, Warehouse Management System, rappresenta il terzo livello gerarchico ed è un sistema informativo per la preparazione, il monitoraggio e l'esecuzione delle attività di magazzino di natura transazionale. Il WMS gestisce la distinta base dividendo le eventuali missioni di picking manuale o automatico. Quest'ultimo è attivato tramite il Warehouse Control

System o WCS che è il secondo livello gerarchico della struttura IT e rappresenta un importante collegamento tra il software di gestione del magazzino (WMS) e le attrezzature di movimentazione dei materiali. Pertanto, il software ha lo scopo di coordinare le operazioni quotidiane all'interno di un centro di magazzino e ottimizza i movimenti del sistema AS/RS applicando l'algoritmo di Dijkstra all'insieme delle locazioni da visitare. Operativamente, il WCS invia i comandi al Controllore Logico Programmabile (PLC), il quale si occupa di gestire tutti gli elementi di movimentazione automatica inviando segnali agli attuatori installati sulla forca e sui nastri trasportatori. Il PLC è lo strumento più operativo dell'infrastruttura IT, tuttavia, non è stato incluso nello sviluppo di questo progetto.

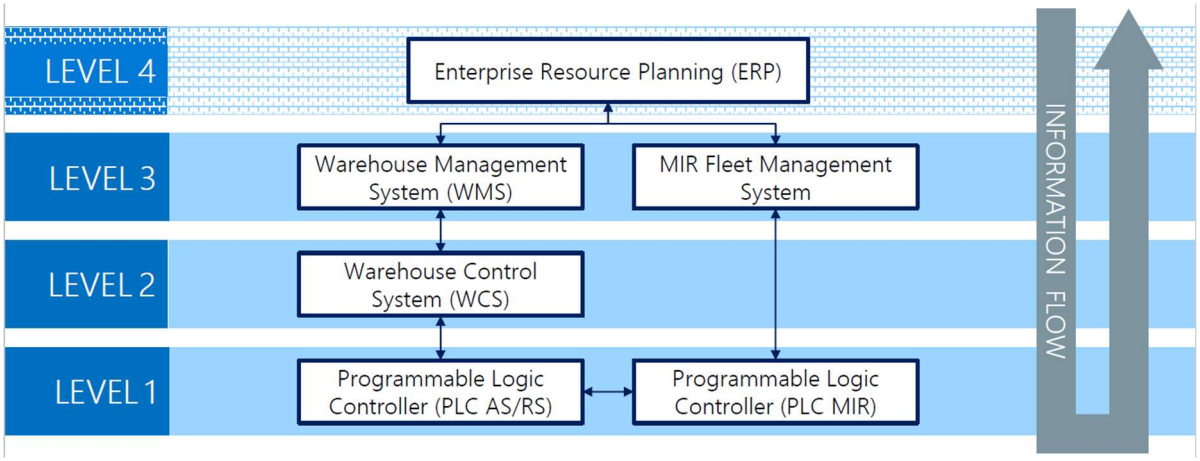


***Figura 5.10** – Livelli sistema informativo-gestionale*

6. Modello Concettuale

La seconda fase della modellazione di simulazione è la costruzione di un modello concettuale. In questo paragrafo verrà analizzato il modello concettuale che illustra il processo preliminare alla realizzazione del Digital Shadow del magazzino automatico *(Figura 6.1).*

In particolare, il modello concettuale include tutti gli elementi che caratterizzano il modello virtuale e la relativa descrizione dei contenuti dell'ambiente modellato; in aggiunta, illustra quali sono le assunzioni e le semplificazioni fatte durante il processo di astrazione.

È stato creato un diagramma di classe attraverso il linguaggio UML (Unified Modelling Language), poiché questo tipo di diagrammi può essere utilizzato per descrivere efficacemente i DS e DT e la connessione tra le varie entità coinvolte (Azangoo et al., 2020).

Il processo logico che si è riprodotto attraverso la modellazione UML è descritto principalmente dalle classi WMS, Task, Mission, Maxi Shuttle, Box, Storage Location.

Quando il magazzino riceve un ordine il primo agente che interviene è il WMS, il quale genera un Task specifico per la richiesta ricevuta e lo assegna ad una Sessione. Il WMS gestisce una sessione alla volta che è composta da un massimo di quattro task che possono coinvolgere fino a quattro Box ciascuno, per un totale di 16 complessive. Il WMS, quindi, ha il compito fondamentale di ottimizzare le operazioni di magazzino per efficientare la movimentazione delle box. Per quel che riguarda il task, esso è caratterizzato dall'informazione relativa alla tipologia di compito da svolgere, prelievo o stoccaggio, e dalla cassetta interessata. I dati del task sono tradotti in missioni operative; anch'esse sono suddivise in missioni di prelievo e missioni di stoccaggio, tuttavia, contengono attributi aggiuntivi come il vano e la posizione occupata al suo interno. Nel sistema reale le missioni servono per la gestione dei movimenti del trasloelevatore che ha la libertà di spostarsi sia orizzontalmente sia verticalmente nella scaffalatura. Per esigenze di modellazione è stato necessario suddividere l'entità trasloelevatore in due classi definite Maxi Shuttle e Shuttle. La prima si occupa di compiere gli spostamenti in orizzontale (asse x), mentre la seconda quelli in verticale (asse z). Un ulteriore aspetto da sottolineare è che lo shuttle svolge anche la funzione di prelievo e stoccaggio delle box, ovvero il movimento in profondità (asse y). Alla luce di questa scelta progettuale, è stata suddivisa, attraverso una relazione di generalizzazione, la classe mission in Maxi Shuttle Mission e Shuttle Mission. Tali classi sono sostanzialmente identiche ma si riferiscono, rispettivamente, al Maxi Shuttle e allo Shuttle.

La classe ASRS rappresenta il magazzino stesso ed è composto dal Maxi Shuttle, il quale a sua volta contiene lo shuttle, e dalle locazioni di stoccaggio identificate nella classe Storage Location. Essa è caratterizzata da differenti informazioni, quali, ad esempio, le coordinate della posizione del vano e il valore assegnato che ne identifica la priorità. Logicamente, lo Storage Location ha lo scopo di ospitare le Box, pertanto nel diagramma UML essi sono collegati attraverso una relazione semplice

e ogni locazione può contenere fino a quattro cassette contemporaneamente. In ultimo, l'entità box è qualificata attraverso gli attributi che ne indicano le dimensioni e il tipo.

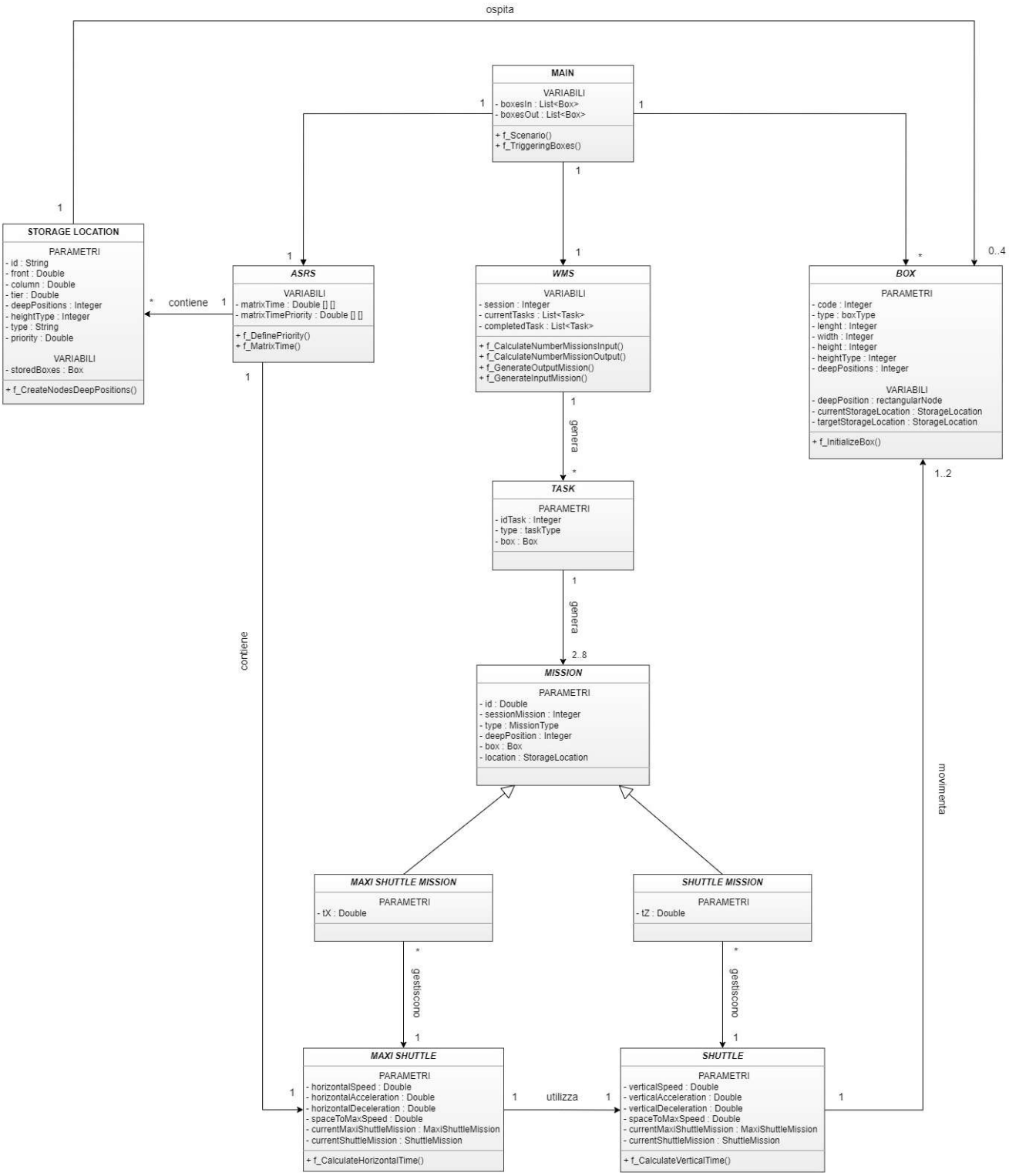


Figura 6.1 – *Diagramma UML del sistema*

7. Modello Virtuale

Dopo aver analizzato il modello reale e aver costruito il modello concettuale, il terzo step per la modellazione di un sistema di simulazione riguarda la realizzazione del modello virtuale.

Il modello è stato sviluppato con Anylogic, un software di simulazione multi-metodo, ovvero caratterizzato dall'integrazione di differenti metodi di simulazione attraverso cui è possibile superare i limiti e gli svantaggi degli approcci considerati singolarmente e attingendo il meglio da ciascuno. Le metodologie principali implementate sono la modellazione agent-based, il paradigma DES e la simulazione system dynamics. La prima, è incentrata sui singoli componenti attivi di un sistema, noti come agenti, di cui devono essere definiti i comportamenti specifici. La seconda, descrive i processi come una sequenza di eventi discreti separati e, in genere, non caratterizza i dettagli fisici. Tale metodologia è ampiamente impiegata nei settori della produzione e della logistica. Infine, vi è la modellizzazione dei sistemi dinamici, un metodo fortemente astratto e principalmente usato per affrontare problemi a livello strategico. Pertanto, ignora i dettagli precisi di un sistema, come, ad esempio, le singole proprietà di persone, prodotti o eventi, e produce una rappresentazione generale di un sistema complesso.

I tre metodi possono essere usati in qualsiasi combinazione, con un unico software, per simulare sistemi aziendali e industriali di qualsiasi complessità.

Un altro punto di forza del software Anyogic è la possibilità di realizzare modelli parametrici ovvero tramite dei parametri degli agenti che possono essere cambiati durante la fase di runtime così da modificare il modello. Questa tipologia di implementazione ha il vantaggio di rendere il modello flessibile con lo scopo di simulare differenti scenari come, ad esempio, nel caso del magazzino in questione, la creazione di nuovi vani o la modifica di caratteristiche tecniche del trasloelevatore per valutare il cambiamento del comportamento del sistema e degli indici di performance (KPI).

Inoltre, Anylogic fornisce un set di librerie e strumenti specifici per diversi settori grazie al quale è possibile modellizzare con diversi livelli di dettaglio.

Gli strumenti principali utilizzati per lo sviluppo del modello del magazzino AS/RS sono:

- Elementi di space markup;
- Libreria di process modelling;
- Libreria sul material handling;
- Elementi aggiuntivi per gli agenti.

1- Elementi di Space Markup

In primo luogo, il software Anylogic offre gli space markup per descrivere l'ambiente e definire la posizione degli agenti nel magazzino. In particolare, essi permettono di animare gli agenti che popolano l'ambiente simulato.
Gli elementi di space markup più utilizzati nella modellizzazione del magazzino sono descritti nella *Tabella 7.1* seguente:

Tabella 7.1 *– Elementi di Space Markup*

Icona	Space markup	Descrizione
	Rectangular Node	Il nodo definisce un'area in cui gli agenti possono risiedere. I nodi rettangolari possono essere collegati con percorsi e formare una rete
	Point Node	Il nodo puntuale, a differenza del rectangular node, non ha un'area, ma solo una dimensione puntuale in cui gli agenti possono risiedere
	Attractor	Gli attrattori consentono di controllare la posizione dell'agente all'interno di un nodo rettangolare o poligonale. Se il nodo definisce la destinazione del movimento dell'agente, gli attrattori definiscono le posizioni esatte all'interno del nodo. Se il nodo definisce la posizione di attesa, gli attrattori definiscono i punti esatti in cui gli agenti aspetteranno all'interno del nodo. Gli agenti si dirigono verso la posizione dell'attrattore per l'attesa
	Conveyor	Conveyor è la forma di markup dello spazio che definisce graficamente un trasportatore

2- Libreria di Process Modelling

Dal momento che gli space markup sono elementi che permettono solo di rappresentare l'aspetto fisico del sistema non sono sufficienti a modellizzare il magazzino. Pertanto, per configurare il processo in termini di agenti, operazioni e risorse, Anylogic offre una libreria di process modeling che permette di creare modelli basati sui processi di un sistema reale. La libreria contiene molti blocchi che permettono di configurare i processi in forma di flowchart. I diagrammi di flusso di Anylogic sono gerarchici, scalabili ed estensibili. Inoltre, sono orientati agli oggetti dal momento che il software è basato sul linguaggio di programmazione java. Grazie a queste caratteristiche, il software permette di modellare sistemi complessi a qualsiasi livello di dettaglio. Infine, la libreria di process modelling permette di animare i processi

attraverso il collegamento con gli elementi di space markup. Gli elementi di process modelling più utilizzati nella modellizzazione del magazzino sono descritti nella *Tabella 7.2* seguente:

***Tabella 7.2** – Elementi di Process Modelling*

Icona	Nome Blocco	Descrizione
	Source	Genera gli agenti
	Sink	Elimina gli agenti
	Delay	Ritarda gli agenti del tempo specificato. Può essere utilizzato per indicare il tempo di processamento di un agente
	Queue	Rappresenta una coda di agenti e permette di stabilire la politica di gestione, come, ad esempio, FIFO o LIFO
	Wait	Questo blocco è simile al blocco Queue, con un'eccezione: supporta il recupero manuale attraverso delle funzioni specifiche; pertanto, non ha un ordinamento
	Select Output	Inoltra l'agente a una delle porte di uscita, a seconda della condizione
	Hold	Blocca il flusso di agenti
	Move To	Sposta un agente dalla sua posizione attuale a quella nuova
	Enter	Inserisce gli agenti (già esistenti) in un punto particolare del flowchart. Da non confondere con il modulo Source
	Exit	Toglie gli agenti in arrivo dal flusso del processo e consente all'utente di specificare cosa fare con essi. Da non confondere con il modulo Sink

3- Libreria sul Material Handling

È offerta da Anylogic anche una libreria specifica per la movimentazione dei materiali. La libreria di material handling è stata impiegata solo per la modellizzazione dei conveyor, tuttavia, permette maggiori possibilità. Gli elementi più utilizzati nella modellizzazione del magazzino sono descritti nella *Tabella 7.3* seguente:

***Tabella 7.3** – Elementi di Material Handling*

Icona	Nome Blocco	Descrizione
	Conveyor Enter	Colloca gli agenti in arrivo su una rulliera, ma non avvia il loro trasporto

	Convey	Trasporta gli agenti al punto di destinazione specificato tramite i trasportatori. È l'unico blocco che controlla il movimento dei materiali all'interno di una rulliera
	Conveyor Exit	Rimuove gli agenti in arrivo da un conveyor e li invia ulteriormente attraverso la porta di uscita come agenti regolari

4- Elementi aggiuntivi per gli agenti

Secondo il paradigma della modellazione ad agenti, un modello è composto da una moltitudine di agenti, con differenti caratteristiche e proprietà, che interagiscono tra di loro per generare un'astrazione di un problema del mondo reale. Per descrivere le caratteristiche degli agenti, sono messi a disposizione da Anylogic degli elementi aggiuntivi rispetto alle precedenti librerie. Gli elementi *"Agent"* più utilizzati nella modellizzazione del magazzino sono descritti nella *Tabella 7.4* seguente:

***Tabella 7.4** – Elementi aggiuntivi per gli agenti*

Elementi	Descrizione
Parameters	I parametri sono generalmente delle costanti e rappresentano caratteristiche particolari di un oggetto all'interno del modello. Diventano fondamentali nella descrizione di istanze diverse dello stesso agente che si differenziano proprio per il valore assunto dal parametro.
Variables	Le variabili rappresentano lo stato del modello e possono cambiare durante la simulazione. Vengono normalmente utilizzate per modellare alcune caratteristiche dinamiche degli oggetti o per memorizzare i risultati della simulazione del modello. Le variabili sono valori di una classe Java o di un tipo scalare arbitrario.
Collections	Un gruppo di oggetti della stessa classe genera una collezione. Le collezioni permettono di definire questo gruppo di elementi in un'unica unità. Per questo motivo, permettono di manipolare dati aggregati. Vari sono i tipi di collezioni utilizzabili in Anylogic, come array-list, linked-list, tree-set, tree-map. Queste si differenziano per la metodologia di accesso e manipolazione dei dati. Nel modello proposto è stato utilizzato solo il primo tipo. In particolare, le liste di array sono costituite da vettori ridimensionabili dotati di una propria capacità. Quando un elemento viene aggiunto all'elenco di array, la capacità aumenta automaticamente.

Elementi	Descrizione
Option List	Gli elenchi di opzioni sono un elemento incorporato del software che viene utilizzato per definire alcuni attributi dell'agente che possono assumere solo particolari valori alternativi o opzioni. Ogni elenco di opzioni è composto da elementi specifici che rappresentano le diverse occorrenze che l'attributo può assumere. Una volta definito, l'elenco di opzioni diventa un tipo assegnabile a parametri o variabili.
Functions	Anylogic consente di definire funzioni proprie. Esse si rivelano utili quando è necessario eseguire qualche operazione difficile da modellare direttamente con gli oggetti e gli elementi incorporati nel software. Un'altra importante potenzialità delle funzioni è la possibilità di utilizzarle in luoghi e momenti diversi del modello. Le funzioni sono in grado di restituire valori specifici come risultato delle stesse o semplicemente di eseguire il codice e di compiere l'azione. Sono scritte in Java ed è possibile sfruttare tutte le peculiarità di questo linguaggio.
Events	Gli eventi sono lo strumento per programmare alcune azioni all'interno del modello. Vengono utilizzati soprattutto quando è necessario ripetere ciclicamente un'operazione. Nel modello si utilizzano eventi ciclici. Dopo un determinato momento, l'evento si verifica in base a una determinata frequenza.

5- State chart

Alcuni agenti, talvolta, presentano dei comportamenti più complessi da descrivere e, quindi, possono essere utili degli elementi aggiuntivi per caratterizzare un diagramma degli stati. Uno state chart è costituito da stati e transizioni ed è fondamentale per rappresentare attività guidate dal tempo e dagli eventi.

Gli stati rappresentano una particolare condizione di un agente in uno specifico momento durante la simulazione, mentre le transizioni consentono il passaggio da uno stato all'altro e possono essere causate da diversi fattori. Quando uno state chart si trova in uno stato particolare, i *trigger* che possono innescare una transizione verso un cambiamento di stato sono generati da una serie di eventi che devono verificarsi. La struttura degli state chart messa a disposizione da Anylogic garantisce che esso sia sempre in attesa che avvenga una di queste condizioni. Le condizioni di transizione degli state chart più utilizzati nella modellizzazione del magazzino sono descritti nella tabella seguente:

Tabella 7.5 – *Elementi dello State Chart*

Tipo	Descrizione
Message	La transizione avviene alla ricezione di un messaggio specifico ricevuto dal diagramma di stato
Agent arrival	La transizione avviene quando l'agente raggiunge il punto di destinazione dopo aver compiuto un movimento
Condition	La transizione avviene quando una data condizione diventa vera. La condizione è un'espressione booleana arbitraria e può dipendere dagli stati di qualsiasi agente dell'intero modello con dinamiche continue e discrete. Nella maggior parte dei casi si può assumere che la condizione sia costantemente monitorata mentre la transizione è attiva

7.1 Struttura del modello

I modelli su Anylogic si fondano su una struttura gerarchica: ogni agente può incapsularne altri generando diversi livelli di profondità e, dunque, creando un albero con differenti ramificazioni. L'agente di più alto livello corrisponde alla radice dell'albero e rappresenta il grado maggiore di astrazione del modello, mentre, gli agenti sottostanti permettono di rappresentare in modo più dettagliato il sistema.

Il modello è costituito da un agente top-level chiamato Main e altri nove agenti di più basso livello. Questi sono:

1. Box;
2. Wms;
3. Task;
4. Asrs;
5. StorageLocation;
6. MaxiShuttle;
7. Shuttle;
8. MaxiShuttleMission;
9. ShuttleMission.

Per una migliore comprensione degli agenti e delle relazioni che li descrivono è stato scelto un approccio top-down. La struttura gerarchica del modello analizzato è mostrata nella seguente *Figura 7.1:*

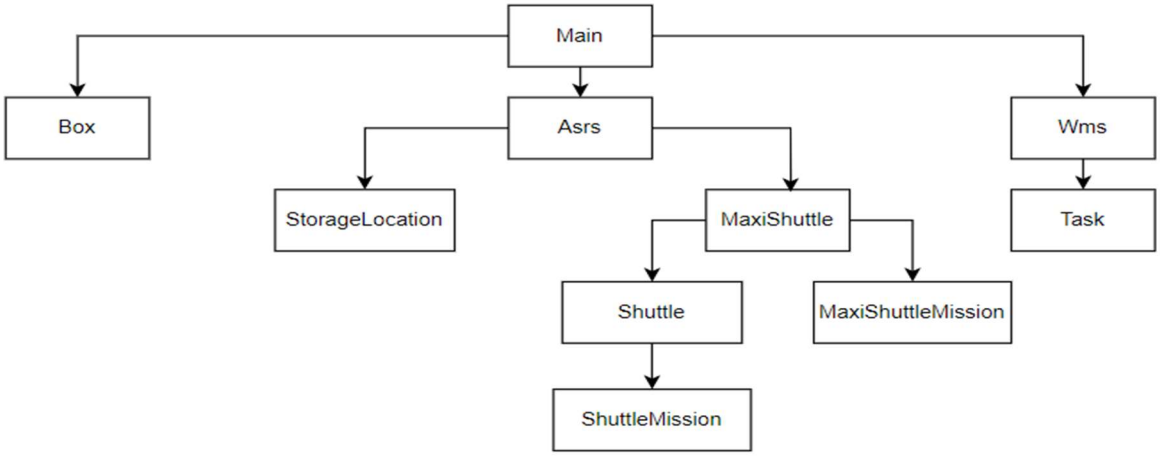


***Figura 7.1** – Struttura gerarchica modello di simulazione*

7.1.1 Main

Il Main è il livello più generale del modello e indica l'ambiente in cui vivono gli agenti principali nonché ulteriori elementi utili durante lo svolgimento della simulazione *(Figura 7.2)*. In particolare, al contrario di altri agenti del modello, il Main non ha nessuna rappresentazione fisica.

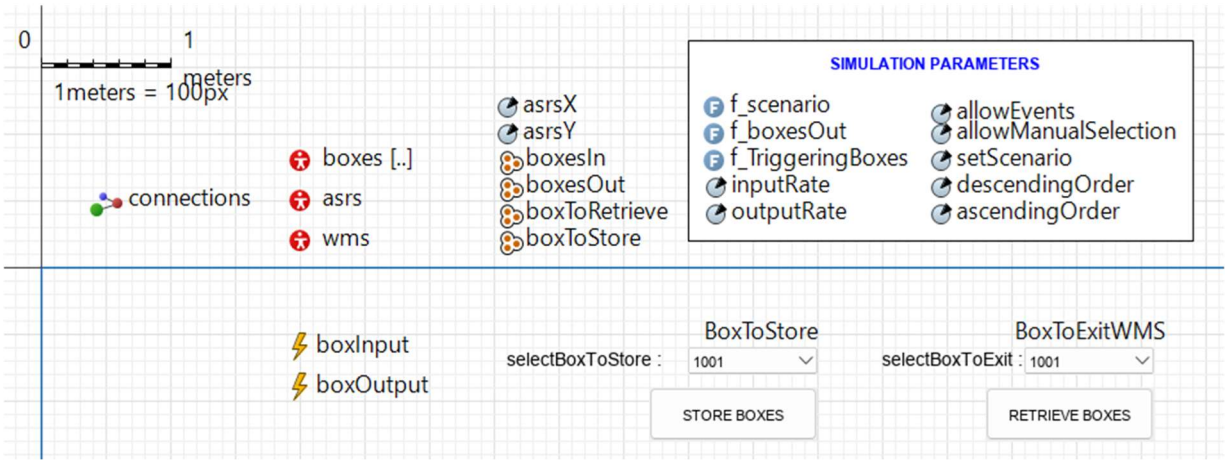


***Figura 7.2** - Main*

Gli attributi che descrivono l'agente Main sono schematizzati nella *Tabella 7.6*, mentre, i parametri e le funzioni relative alle impostazioni della simulazione sono presenti nella *Tabella 7.7*.

In primo luogo, vi sono tre differenti agenti che sono incapsulati nel Main: la popolazione Boxes le cui istanze sono gli agenti Box, l'agente Asrs che rappresenta il magazzino automatico e, infine, l'agente Wms relativo alla gestione operativa dei Task. Questi agenti sono descritti approfonditamente più avanti nell'analisi.

In secondo luogo, i due parametri asrsX e asrsY sono utilizzati per impostare, all'avvio della simulazione, la posizione iniziale dell'agente Asrs.

Infine, le quattro collezioni boxesIn, boxesOut, boxToRetrieve e boxToStore rappresentano dei vettori designati a contenere informazioni utili a organizzare correttamente le Box del magazzino. In particolare, le prime due collezioni servono a indicare, rispettivamente, quali box sono immagazzinate nella scaffalatura e quali no, mentre, le altre due contengono i codici delle cassette che devono essere prelevate o stoccate.

Gli elementi più pratici presenti nel Main sono i due eventi e i due tasti, essi sfruttano le informazioni contenute dalle precedenti quattro collezioni. Gli eventi, denominati boxInput e boxOutput, generano ciclicamente ordini di stoccaggio e di prelievo; il tasso che caratterizza il verificarsi di un evento è impostato, in fase preliminare, assegnando un valore ai parametri inputRate e outputRate. I tasti svolgono un ruolo analogo seppur con la differenza che gli ordini di input e output sono inviati manualmente da chi svolge l'esperimento.

Con riferimento alla *Figura 7.2* - Main, gli attributi del Main definiti come "Simulation Parameters" rappresentano ulteriori meccanismi svolti dall'agente Main; essi sono spiegati nel paragrafo successivo *Simulation.*

Tabella 7.6 – *Attributi Main*

Nome	Elemento	Tipo	Descrizione
asrsX	Parametro	double	Coordinata x dell'agente Asrs all'interno del Main
asrsY	Parametro	double	Coordinata y dell'agente Asrs all'interno del Main
boxesIn	Collezione	ArrayList<Box>	Collezione delle box all'interno del magazzino
boxesOut	Collezione	ArrayList<Box>	Collezione delle box all'esterno del magazzino
boxesToRetrieve	Collezione	ArrayList<int>	Collezione dei codici delle box da prelevare
boxesToStore	Collezione	ArrayList<int>	Collezione dei codici delle box da stoccare

Nome	Elemento	Tipo	Descrizione
f_boxesOut	Funzione	-	Per aggiungere le box alla collezione boxesOut

Nel Main è presente anche un flow chart per la gestione delle cassette in input al magazzino automatico *(Figura 7.3)*. Il flusso seguito da ciascuna Box per entrare nel sistema inizia nel blocco *enter*: gli agenti sono inseriti a seguito delle funzioni svolte dagli eventi e dai tasti descritti sopra. Successivamente il modulo *controllFirstTime* suddivide il flusso in due rami distinti a seconda del fatto che la Box in questione sia coinvolta per la prima volta in una operazione di magazzino o meno; in particolare, questa soluzione garantisce che dopo l'uscita di una box dal magazzino, l'agente rimane in attesa di una nuova missione di input nel blocco *wait*. Seguendo il flusso, esso si ricongiunge in una coda di Box, gestita con logica FIFO. Infine, a termine del flow chart, è presente un blocco exit, denominato *toAsrs,* che ha la funzione di trasportare le box in un un'altra parte del modello, più precisamente nell'agente Asrs.

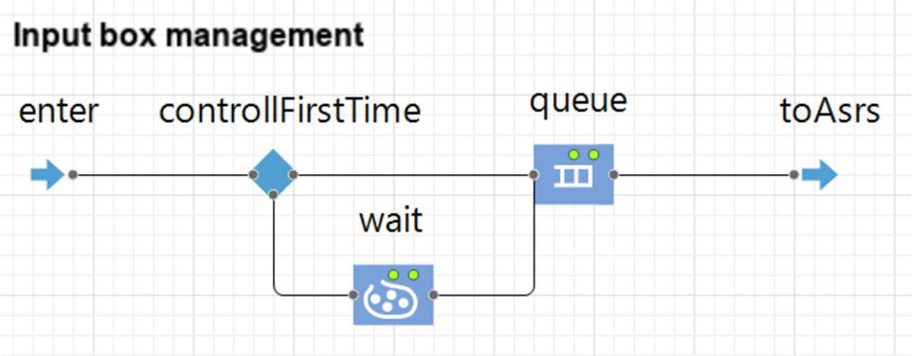


Figura 7.3 – *Flowchart Main*

Simulation

Il software Anylogic permette di impostare, all'avvio di ogni simulazione, una pagina iniziale in cui è possibile settare dei parametri del modello personalizzati e degli esperimenti *(Figura 7.4)*. In questo modo, lo stesso modello può essere studiato più approfonditamente in differenti scenari di simulazione.

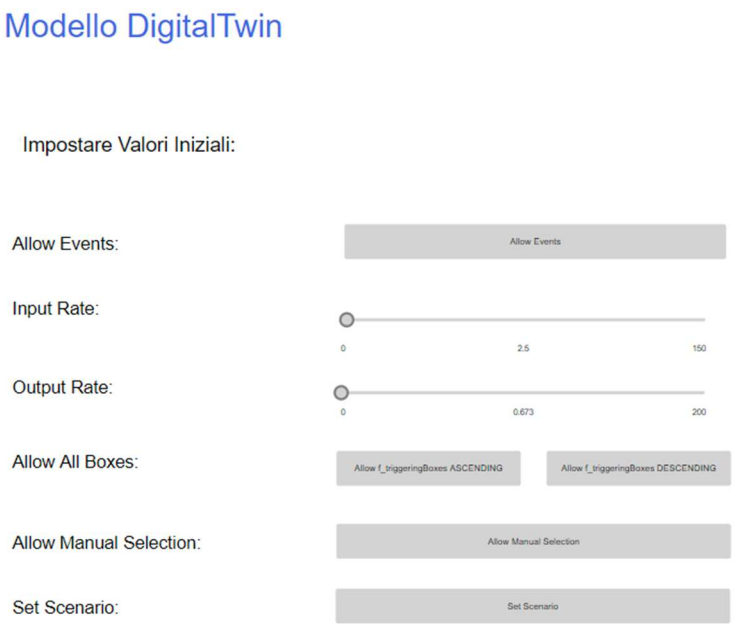


***Figura 7.4** – Pagina iniziale run di simulazione*

Gli attributi descritti nella *Tabella 7.7* – Impostazioni simulazione sono definiti attraverso l'utilizzo dei comandi a disposizione nella pagina iniziale della simulazione.

***Tabella 7.7** – Impostazioni simulazione*

Nome	Elemento	Tipo	Descrizione
inputRate	Parametro	double	Tasso di domanda di Box richieste in input
outputRate	Parametro	double	Tasso di domanda di Box richieste in output
allowEvents	Parametro	int	Indica se gli eventi sono abilitati o meno
allowManualSelection	Parametro	int	Permette la selezione manuale delle box da prelevare o stoccare
setScenario	Parametro	int	Permette di impostare uno scenario di partenza del magazzino
descendingOrder	Parametro	int	Imposta l'ordine decrescente per la funzione f_TriggeringBoxes

Nome	Elemento	Tipo	Descrizione
ascendingOrder	Parametro	int	Imposta l'ordine crescente per la funzione f_TriggeringBoxes
f_scenario	Funzione	-	Imposta lo stato iniziale del magazzino
f_TriggeringBoxes	Funzione	-	Invia in input al magazzino tutte le box secondo un ordine stabilito

7.1.2 Box

In *Figura 7.5* è mostrata una vista del modello relativa all'agente Box.

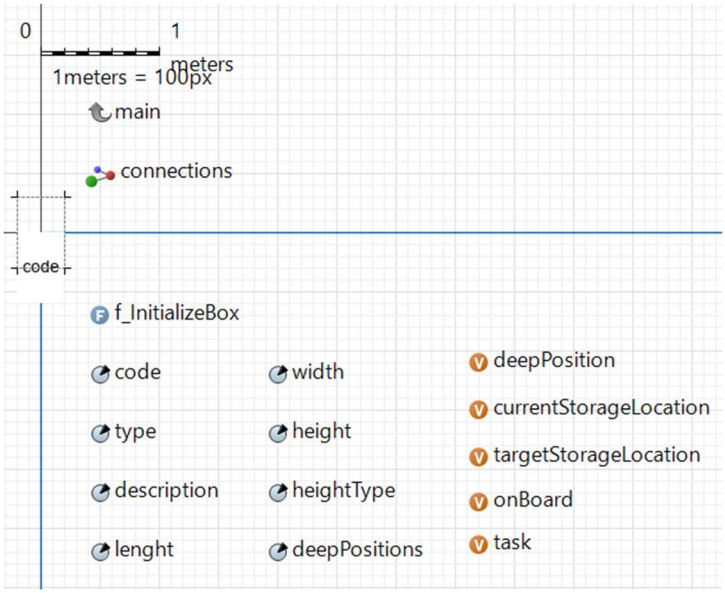


Figura 7.5 - *Box*

L'agente Box rappresenta le cassette che sono movimentate all'interno del magazzino. Esse possono essere di cinque diverse tipologie *(Tabella 5.1)* ma non costituiscono agenti diversi bensì cinque istanze della stessa popolazione di agenti. Questa scelta di modellazione è stata effettuata a causa delle differenze parziali tra le differenti Box; infatti, le variazioni non comprendono diagrammi di stato o altri elementi rilevanti ma solo differenti attributi dell'agente *(Tabella 7.8).*

Tabella 7.8 – *Attributi Box*

Nome	Elemento	Tipo	Descrizione
code	Parametro	int	Codice univoco per l'identificazione
type	Parametro	boxType	Specifica il tipo (LTB4120, LTB4220, LTB6120, LTB 6220, MF6070)
description	Parametro	String	Indica la grandezza (big o small)
length	Parametro	int	Indica la lunghezza
width	Parametro	int	Indica la larghezza
height	Parametro	int	Indica l'altezza
heightType	Parametro	int	Indica il tipo di altezza (1==bassa o 2==alta)
deepPositions	Parametro	int	Posizioni occupate (1 o 2)
deepPosition	Variabile	RectangularNode	Indica la locazione del vano in cui la Box è stoccata
currentStorageLocation	Variabile	StorageLocation	Indica il vano corrente
targetStorageLocation	Variabile	StorageLocation	Indica il vano di destinazione
task	Variabile	task	Indica il task in cui appartiene la Box
f_InizializeBox	Funzione	Just Action	Permette l'inizializzazione delle Box. In particolare, imposta le dimensioni descritte dai parametri e assegna un colore in base alla tipologia (type).

Grazie al software Anylogic e alla potenzialità della simulazione parametrica è possibile estrapolare i differenti attributi di ogni Box tramite query SQL dalla tabella Excel denominata *Boxes*. In essa sono presenti tutte le cassette esistenti in magazzino con tutte le caratteristiche necessarie per descriverle.

All'interno del modello, l'elemento Box è rappresentato da un parallelepipedo di dimensione variabile in base alla grandezza della cassetta e per differenziare le tipologie sono stati usati dei colori, in particolare, il giallo è associato alle cassette di tipo LTB6120, l'arancione alla tipologia LTB6220, il grigio alla tipologia LTB4120, il verde alla tipologia LTB4220 e, infine, l'azzurro per il vassoio MF6070.

7.1.3 WMS

Il WMS è l'agente che si occupa di gestire le operazioni all'interno del magazzino. In primo luogo, il WMS ospita e amministra la popolazione di agenti Task che rappresenta l'insieme di tutti i compiti del magazzino che, a loro volta, hanno lo scopo di inviare le missioni operative al Maxi Shuttle e allo Shuttle. In *Figura 7.6* è mostrata una vista di Anylogic relativa al WMS, mentre gli attributi sono descritti nella *Tabella 7.9* – Attributi WMS.

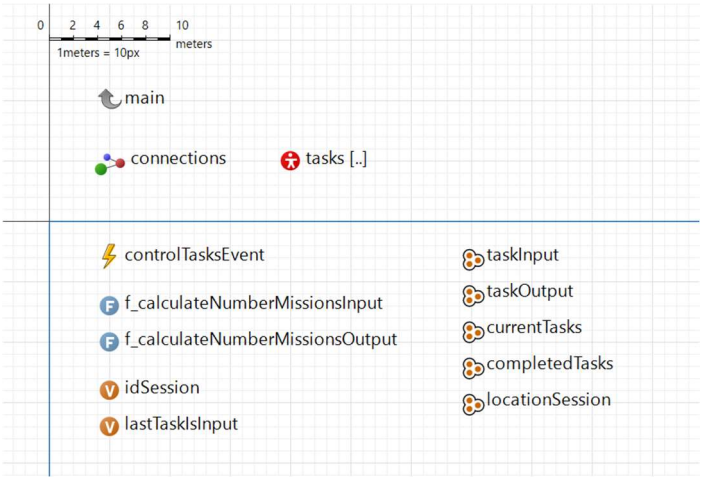


***Figura 7.6** – WMS*

***Tabella 7.9** – Attributi WMS*

Nome	Elemento	Tipo	Descrizione
idSession	Variabile	int	Codice univoco per l'identificazione
lastTaskInput	Variabile	boolean	Identifica il tipo di task gestito nella sessione precedente

Nome	Elemento	Tipo	Descrizione
taskInput	Collezione	ArrayList<Task>	Collezione di task relativi all'input presenti in coda
taskOutput	Collezione	ArrayList<Task>	Collezione di task relativi all'output presenti in coda
currentTasks	Collezione	ArrayList<Task>	Collezione di task gestiti nella sessione corrente
completedTasks	Collezione	ArrayList<Task>	Collezione di task completati
locationSession	Collezione	ArrayList<StorageLocation>	Collezione di vani relativi alle Box della sessione corrente
controlTasksEvent	Evento	Timeout-Cyclic	Esegue un controllo per gestire l'avvio della sessione di input/output
f_calculateNumber MissionsInput	Funzione	-	Calcola i Task da assegnare ad una sessione di input
f_calculateNumber MissionsOutput	Funzione	-	Calcola i Task da assegnare ad una sessione di output

Il flow chart in *Figura 7.7* rappresenta come è stata modellizzata la gestione. I Task giungono nel flusso attraverso il blocco *Enter* e si accodano nel modulo successivo in cui sono assegnati a differenti collezioni in base alla tipologia di compito, ovvero di stoccaggio o di recupero. Il WMS svolge delle sessioni di input o di output, composte da quattro task ciascuna, e le alterna ciclicamente. Uno degli elementi principali per realizzare questo funzionamento è l'evento *controlTasksEvent* che ogni 500 millisecondi verifica che ci siano elementi nella coda *arriveQueueWms* e, se presenti, aggiorna la variabile booleana relativa all'ultima tipologia di sessione svolta e chiama la funzione necessaria a far proseguire le entità corrette nel flusso. I task permangono nel modulo *Delay* fino a che la Box connessa al Task non raggiunge il nastro trasportatore in uscita per i Task di output o il deposito nel vano per i Task di input. Infine, il flusso si conclude con il blocco *Exit* in cui è popolata la collezione relativa ai *completedTasks* e sono ripristinate le impostazioni per ricominciare una nuova sessione.

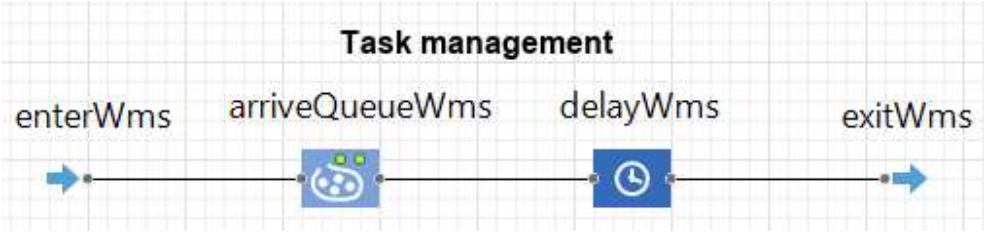


***Figura 7.7** – Flowchart WMS*

7.1.4 Task

L'agente Task, a differenza di altri agenti, non presenta un aspetto fisico ma è introdotto nel modello di simulazione per gestire al meglio i compiti del magazzino *(*Figura 7.8*)*. In particolare, l'agente è costituito da un esiguo numero di parametri, tra cui uno che identifica la Box interessata alla movimentazione e uno che definisce il tipo di compito: storage o retrieval *(Tabella 7.10 –* Attributi Task*)*. Ogni qual volta che è richiesta una box all'output o si seleziona una box all'input, si genera un task relativo alla cassetta coinvolta che successivamente sarà gestito all'interno dell'agente WMS.

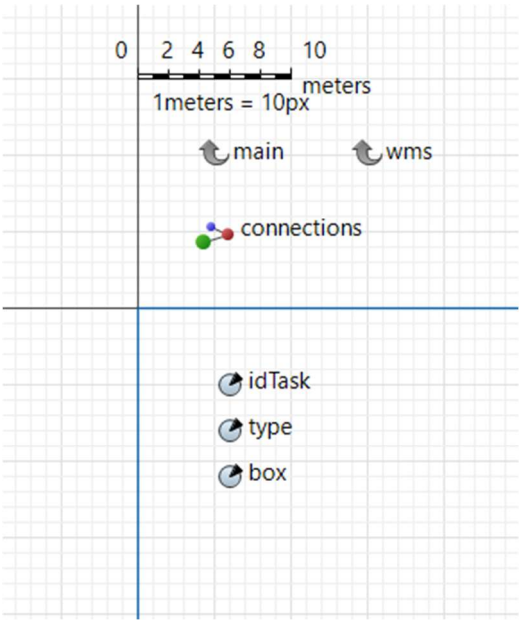


***Figura 7.8** – Task*

Tabella 7.10 – *Attributi Task*

Nome	Elemento	Tipo	Descrizione
idTask	Parametro	int	Codice univoco per l'identificazione
type	Parametro	taskType	Indica il tipo di task (input o output)
box	Parametro	Box	Identifica la Box soggetta al task

7.1.5 Asrs

L'agente Asrs è uno degli agenti principali del modello e racchiude diversi elementi al suo interno, fra cui, la rappresentazione grafica del magazzino, gli attributi, le funzioni e due differenti flow chart.

Dalla *Figura 7.9* - Asrs si nota che gli elementi grafici presenti sono i due nastri trasportatori dedicati all'ingresso e all'uscita dal magazzino, il trasloelevatore e il vano di locazione. In particolare, gli ultimi due elementi appartengono, rispettivamente, all'agente Maxi Shuttle e StorageLocation; tuttavia, sono graficamente visibili dal momento che sono contenuti all'interno dell'Asrs e consentono una migliore interpretazione del magazzino in generale. Infine, sfruttando le possibilità messe a disposizione del Software Anylogic tali blocchi generano sia una rappresentazione in 2D sia una in 3D per una migliore esperienza di simulazione.

Nella *Tabella 7.11* – Attributi Asrs sono descritti i principali attributi dell'agente Asrs.

In primo luogo, all'avvio della simulazione è lanciata la funzione *f_matrixTime* che è necessaria per mappare i tempi da una locazione all'altra. In pratica, è considerato il massimo valore tra il tempo orizzontale e verticale, calcolati attraverso le formule del moto, per la movimentazione del trasloelevatore tra due vani. Iterando ciclicamente questo processo tra tutte le locazioni, si genera la matrice *matrixTime* di dimensioni pari all'intera scaffalatura; inoltre, è utilizzata per condividere i suoi valori con le altre funzioni dell'Asrs. Nello sviluppo di questa funzione, è stata assunta una semplificazione per le coordinate dei vani; in dettaglio, è stata interpretata la scaffalatura come composta da un'unica tipologia di altezza di locazione nonostante il magazzino sia caratterizzato da vani alti e bassi. Stessa semplificazione è effettuata da Incas per il calcolo dei tempi.

In secondo luogo, i valori della *matrixTime* sono utilizzati dalla funzione *f_definePriority* che assegna a ciascun vano un valore di priorità determinato come l'inverso di una media ponderata

tra il numero di ingressi e di uscite. La priorità indica l'ordine di selezione dei vani per le missioni di input che sono raccolti nella collezione *c_storageLocationOrdered*.

Un ulteriore insieme di funzioni sono utilizzate per generare le differenti missioni svolte dal magazzino come, ad esempio, la *f_RetrievalMission*, la *f_StorageMission* e la *f_storageNN.*

Per amministrare gli ordini di input, il *delay* del WMS attiva la funzione *f_generateInputMission* che, a sua volta, richiama la *f_RetrievalMission* per il prelievo della box dal conveyor di ingresso e, successivamente, la *f_StorageMission* per lo stoccaggio della box nel vano disponibile di maggiore priorità tenendo in considerazione che ciascuna udc può essere stoccata solo in vani con dimensioni congruenti.

Viceversa, per gli ordini di output, dal WMS è attivata la funzione *f_generateOutputMission* che, dapprima, valuta se davanti alla box interessata vi sono una o più cassette per cui è necessario compiere movimentazioni interne e, in tal caso, richiama la funzione *f_storageNN.* Quest'ultima si occupa di assegnare il vano di stoccaggio non sulla base della priorità bensì seguendo un principio di *nearest neighbor,* ovvero, è selezionata la locazione disponibile più vicina. Altrimenti sono chiamate in successione la funzione *f_RetrievalMission* e la *f_StorageExitMission* che prelevano la box e la depositano nel conveyor di uscita.

Infine, i parametri settano la posizione del MaxiShuttle all'interno dell'Asrs.

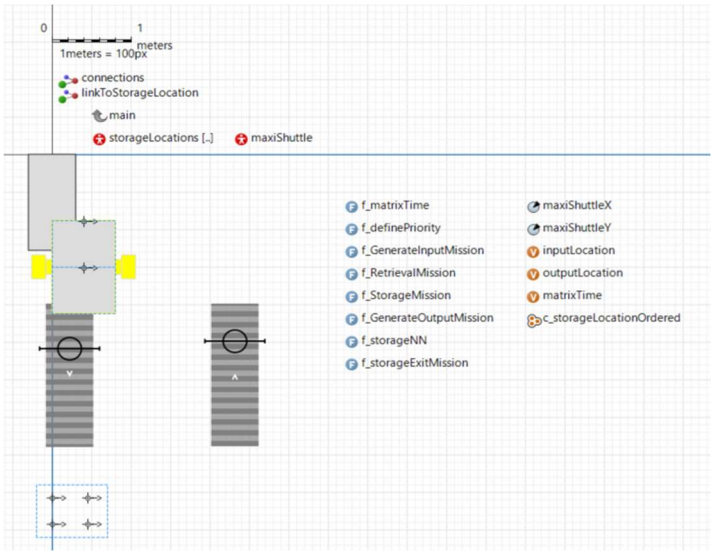


Figura 7.9 - *Asrs*

Tabella 7.11 – *Attributi Asrs*

Nome	Elemento	Tipo	Descrizione
maxiShuttleX	Parametro	double	Indica la posizione del MaxiShuttle lungo l'asse X
maxiShuttleY	Parametro	double	Indica la posizione del MaxiShuttle lungo l'asse Y
inputLocation	Variabile	StorageLocation	Indica il vano di input
outputLocation	Variabile	StorageLocation	Indica il vano di output
matrixTime	Variabile	double [][]	Matrice dei tempi di percorrenza del MaxiShuttle tra i differenti vani
c_storageLocation Ordered	Collezione	ArrayList<StorageLocation>	Collezione di Storage Location ordinati in base alla priorità
f_matrixTime	Funzione	-	Mappa i tempi da un vano all'altro
f_definePriority	Funzione	-	Assegna a ciascun vano un valore di priorità
f_GenerateInput Mission	Funzione	-	Funzione composta dalla f_RetrievalMission e dalla f_StorageMission. La prima consente il prelievo dal vano di input e la seconda stocca la Box nel vano
f_RetrievalMission	Funzione	-	Funzione che permette il prelievo della Box dal vano
f_StorageMission	Funzione	-	Funzione che stocca la Box nel vano
f_GenerateOutput Mission	Funzione	-	Funzione composta dalla f_storageNN, f_retrievalMission e f_storageExitMission.
f_StorageNN	Funzione	-	Identifica un vano per effettuare un'apertura ovvero uno spostamento di

Nome	Elemento	Tipo	Descrizione
			Box che si trovano davanti alla Box richiamata in uscita
f_StorageExit Mission	Funzione	-	Funzione che permette l'uscita dal vano di output delle Box

L'Asrs è composto da due differenti flowchart che descrivono rispettivamente il trasporto della box sul conveyor di input e quello di output *(Figura 7.10 –* Flowchart Asrs*).*

Il flusso di input inizia con il blocco *fromMain*, il quale, come suggerisce il nome, riceve l'agente Box in uscita dal diagramma di flusso del Main. Successivamente, l'agente entra in una serie di blocchi delimitati dai moduli *restrictedArea* affinché non vi siano più di tre box contemporaneamente sul conveyor. Il blocco *convey* trasporta la box dall'ingresso del conveyor fino alla posizione target definita dal *positionOnConveyor,* si tratta di un elemento grafico che identifica la posizione nel conveyor. Una volta giunta la Box al blocco *hold1*, il nastro trasportatore si ferma e verifica che l'ultimo tratto del conveyor sia libero in modo tale che la cassetta possa procedere sino alla fine attraverso il blocco *convey2*. Infine, la Box attende lo sblocco del modulo *hold* generato dall'arrivo del trasloelevatore presso il nastro trasportatore, per essere caricata sullo Shuttle. In particolare, il modulo *hold* è sbloccato quando l'agente MaxiShuttle raggiunge lo stato *loadingUnloading.* Il flusso termina con il blocco *exit* che invia la Box nel flowchart dello Shuttle permettendo ad una nuova Box di occupare una posizione sul conveyor.

Il flusso di output, riguarda, invece, il conveyor di uscita. Questa volta l'agente Box arriva dal diagramma di flusso dello Shuttle ed è inserito nel blocco *enter*. Il modulo *convey1* permette di movimentare la Box sino al termine del conveyor di output e attraverso il blocco *conveyorExit* è rimossa dal nastro trasportatore. È importante sottolineare come i Task di output terminano nel momento in cui lo Shuttle deposita le cassette nel conveyor di uscita, mentre, le collezioni *boxesIn* e *boxesOut* sono aggiornate una volta che le cassette lasciano il conveyor. A seguire, il blocco *moveTo* indica la posizione che le Box devono raggiungere fisicamente una volta uscite dal magazzino. Infine, vi è il modulo *exit*, che riporta l'agente Box nel flusso all'interno del Main in attesa di nuove missioni.

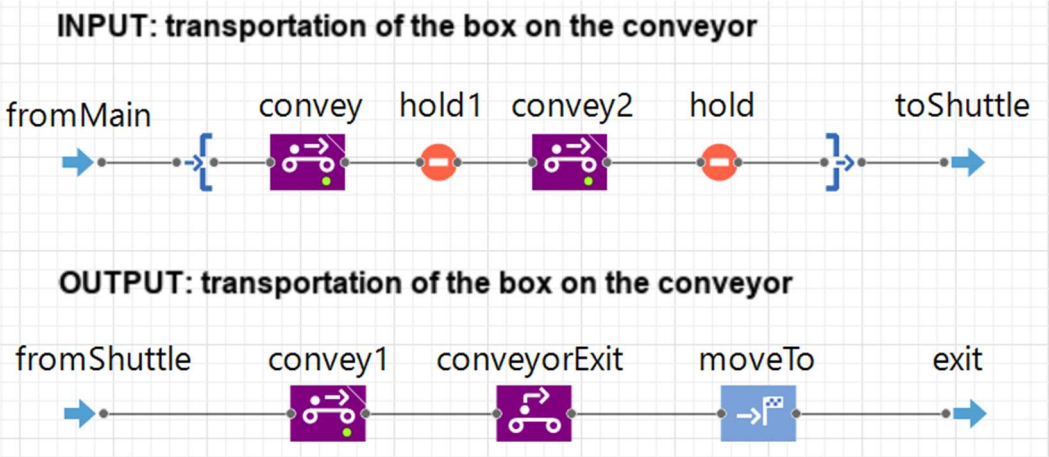


***Figura 7.10** – Flowchart Asrs*

7.1.6 MaxiShuttle

Il MaxiShuttle è uno degli elementi più importanti del magazzino automatico e identifica il trasloelevatore del sistema reale. Più precisamente, è stato scelto di modellizzare la macchina attraverso due agenti differenti per rappresentare, con maggiore semplicità, i movimenti lungo l'asse X e lungo l'asse Z.
Il MaxiShuttle è l'agente che gestisce il movimento orizzontale mentre lo Shuttle si occupa degli spostamenti verticali e sarà analizzato nel prossimo paragrafo.

Da una parte, i parametri dell'agente ne specificano i dati tecnici come, ad esempio, la velocità, l'accelerazione e la decelerazione, dall'altra, le variabili indicano le informazioni relative alla missione corrente, alle box a bordo e all'ultima posizione visitata *(Figura 7.11* - MaxiShuttle*)*.

Per il calcolo del tempo necessario a compiere gli spostamenti orizzontali tra i vani, il MaxiShuttle si serve della funzione *f_CalculateHorizontalTime*. Essa si divide in due parti: la prima, si verifica quando lo spostamento del traslo è minore rispetto allo spazio per raggiungere la massima velocità e calcola il tempo mediante il moto uniformemente accelerato/decelerato; la seconda, rappresenta il caso in cui lo spostamento è maggiore e, quindi, include anche il moto uniforme.

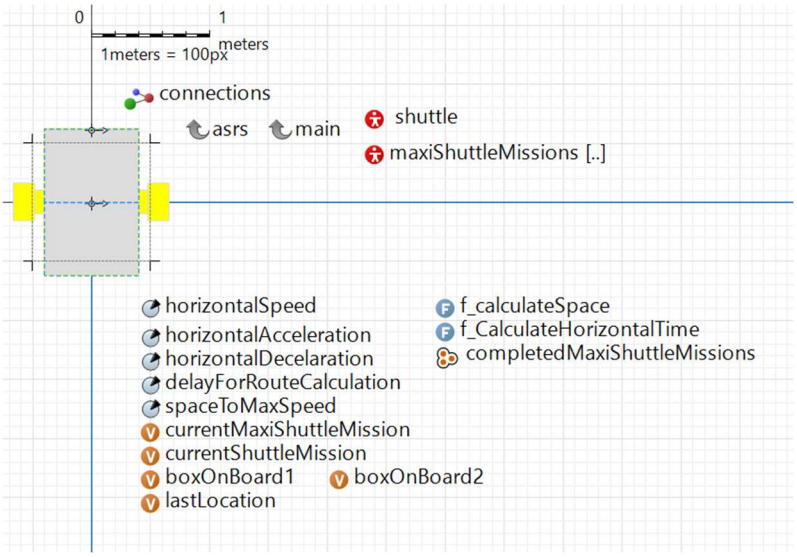


Figura 7.11 - *MaxiShuttle*

Tabella 7.12 – *Attributi MaxiShuttle*

Nome	Elemento	Tipo	Descrizione
horizontalSpeed	Parametro	double	Velocità orizzontale
Horizontal Acceleration	Parametro	boolean	Accelerazione orizzontale
Horizontal Deceleration	Parametro	boolean	Decelerazione orizzontale
delayForRoute Calculation	Parametro	Time [second]	Tempo che impega il MaxiShuttle per calcolare il percorso
spaceToMaxSpeed	Parametro	double	Spazio che il MaxiShuttle deve percorrere per raggiungere la massima velocità orizzontale
currentMaxiShuttleMission	Variabile	MaxiShuttleMission	Indica la missione corrente che il MaxiShuttle sta eseguendo
currentShuttle Mission	Variabile	ShuttleMission	Indica la missione corrente che lo Shuttle sta eseguendo
lastLocation	Variabile	StorageLocation	Ultima posizione raggiunta dal trasloelevatore

Nome	Elemento	Tipo	Descrizione
completedMaxiShuttle Missions	Collezione	ArrayList <MaxiShuttleMission>	Collezione di tutte le missioni completate dal MaxiShuttle
f_calculateSpace	Funzione	-	Calcola lo spazio necessario al MaxiShuttle per raggiungere la massima velocità
f_CalculateHorizontal Time	Funzione	-	Calcola il tempo necessario a spostarsi lungo l'asse orizzontale

Il MaxiShuttle può assumere quattro possibili stati schematizzati nello statechart in *Figura 7.12.* Inizialmente, il MaxiShuttle si trova nello stato *idle*, ovvero di inattività; pertanto, è fermo e in attesa di eseguire una missione di storage o retrieval.
Lo stato successivo è il *moving* e rappresenta il movimento orizzontale del MaxiShuttle verso la posizione della Box oggetto della missione o verso uno StorageLocation. Questo movimento avviene mediante una funzione messa a disposizione da Anylogic, denominata *moveToInTime*, in cui è semplicemente necessario inserire come parametri le coordinate x, y e z di destinazione, e il cosiddetto tripTime, cioè il tempo di viaggio.
La transizione dallo stato *idle* allo stato *moving* avviene quando l'agente MaxiShuttle riceve il messaggio *seized*. Tale messaggio è inviato nel modulo *delay2* del flowchart che descrive la gestione delle missioni assegnate al MaxiShuttle stesso.

Il terzo stato che può assumere l'agente è quello di *arrived* in cui il trasloelevatore ha raggiunto la sua destinazione. In questo stato sono chiamate due funzioni per sbloccare due moduli *hold* relativi ai flowchart del MaxiShuttle e dello Shuttle; tale operazione ha lo scopo di sincronizzare i movimenti e le operazioni svolte da quest'ultimi. La condizione di transizione per raggiungere il terzo stato è di tipo *agent arrival*, ovvero è attivata nel momento in cui il MaxiShuttle raggiunge un punto specifico.

La terza transizione avviene quando il MaxiShuttle riceve il messaggio *loadunload* inviato in ingresso al blocco *delay1* appartenente al flowchart dell'agente stesso. Lo stato raggiunto è denominato *loadingUnloading* in cui il trasloelevatore svolge le operazioni di prelievo o deposito della Box a bordo. In particolare, se sta eseguendo una missione di retrieval da un vano allora è chiamata una funzione affinché la Box sia liberata dal modulo *wait* presente nel diagramma di flusso dello StorageLocation in questione.

Lo statechart ritorna allo stato iniziale di inattività quando riceve il messaggio *released*, inviato in uscita al modulo *delay1*. In pratica, il trasloelevatore segnala di aver concluso l'operazione in corso e, dunque, raggiunge lo stato *idle.*

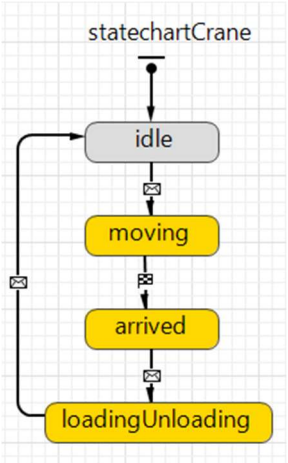


***Figura 7.12** – State chart MaxiShuttle*

L'agente MaxiShuttle sfrutta le MaxiShuttleMission per compiere gli spostamenti; la gestione di quest'ultime avviene attraverso il flowchart in *Figura 7.13*.

Il flusso si avvia nel blocco *enterMission* in cui sono inserite le missioni generate dall'agente Asrs che, successivamente, raggiungono la coda *queue2* dove sono gestite con logica FIFO. Inoltre, in questo modulo è richiamata una funzione di Anylogic per sincronizzare il flusso con quello relativo alla gestione delle missioni dello Shuttle.

Affinché la MaxiShuttleMission possa raggiungere il *delay2* per la presa in carica, è necessario che il modulo hold sia sbloccato e permetta il passaggio di un agente. In particolare, è possibile attivare o disattivare l'hold mediante uno specifico comando di Anylogic.

Nel *delay2* si richiama la funzione *f_CalculateHorizontalTime* per il calcolo del tempo della missione, si aggiornano le informazioni relative alla missione attuale e, infine, il MaxiShuttle riceve il messaggio *seized* che genera un cambiamento di stato da *idle* a *moving* consentendo l'avvio della movimentazione.

I successivi *holdMaxiShuttle* e *holdShuttle* sono sbloccati quando il trasloelevatore si trova nello stato *arrived* ovvero all'arrivo, rispettivamente, del MaxiShuttle e dello Shuttle presso la destinazione della missione.

Infine, una volta raggiunto il *delay1* la missione giunge al termine, il MaxiShuttle passa dapprima allo stato *loadingUnloading* e, successivamente, torna ad essere inattivo. Inoltre, la conclusione di una missione sblocca il modulo *hold* consentendo l'avvio di una eventuale missione in coda.

Il diagramma di flusso termina con il blocco *exit2* in cui la missione è aggiunta alla collezione completedMaxiShuttleMission.

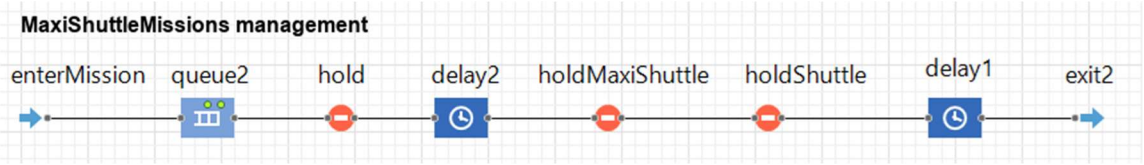


***Figura 7.13** – Flowchart MaxiShuttle*

7.1.7 Shuttle

Lo Shuttle è un agente simile al MaxiShuttle ed è, pertanto, descritto da analoghi attributi presentati *Tabella 7.13*. Tuttavia, è necessario ricordare che lo Shuttle si occupa degli spostamenti verticali. L'elemento di maggiore importanza di questo agente è il flowchart relativo al caricamento delle Box sulla culla del trasloelevatore.

In **Figura 7.14** – Shuttle è presentata una vista dell'agente Shuttle, mentre, per quel che riguarda lo state chart, si rimanda alla *Figura 7.12* e alla relativa descrizione poiché è analogo al MaxiShuttle.

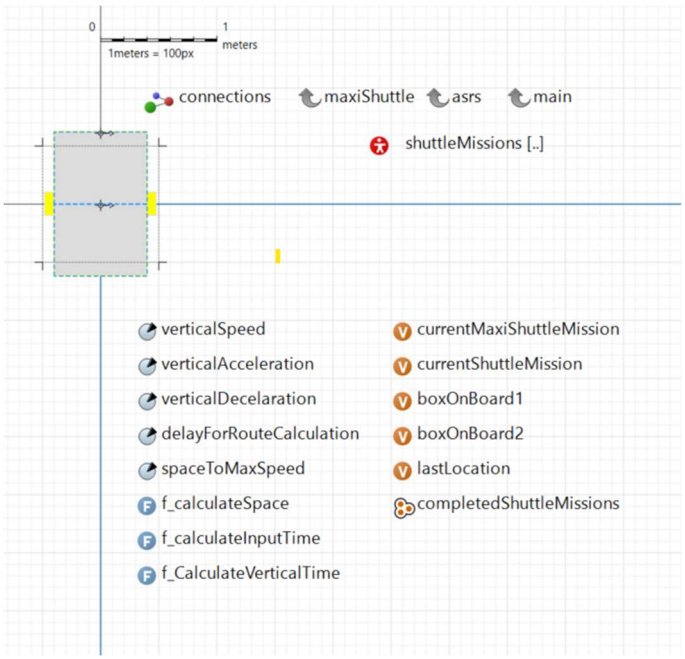


***Figura 7.14** – Shuttle*

Tabella 7.13 – *Attributi Shuttle*

Nome	Elemento	Tipo	Descrizione
verticalSpeed	Parametro	double	Velocità verticale
verticalAcceleration	Parametro	boolean	Accelerazione verticale
verticalDeceleration	Parametro	boolean	Decelerazione verticale
delayForRoute Calculation	Parametro	Time [seconds]	Tempo che impega il Shuttle per calcolare il percorso
spaceToMaxSpeed	Parametro	double	Spazio che il MaxiShuttle deve percorrere per raggiungere la massima velocità verticale
currentMaxiShuttle Mission	Variabile	MaxiShuttleMission	Indica la missione corrente che il MaxiShuttle sta eseguendo
currentShuttleMission	Variabile	ShuttleMission	Indica la missione corrente che lo Shuttle sta eseguendo
boxOnBoard1	Variabile	Box	Box situata sullo Shuttle in posizione 1 (lato fronte 1)
boxOnBoard2	Variabile	Box	Box situata sullo Shuttle in posizione 2 (lato fronte 2)
lastLocation	Variabile	StorageLocation	Ultima posizione raggiunta dal traslo
completedMaxiShuttle Missions	Collezione	ArrayList <MaxiShuttleMission>	Collezione di tutte le missioni completate dallo Shuttle
f_calculateSpace	Funzione	-	Calcola lo spazio necessario allo Shuttle per raggiungere la massima velocità

Nome	Elemento	Tipo	Descrizione
f_calculateInput Time	Funzione	-	Assegna i tempi fissi da considerare in relazione al tipo di box
f_CalculateVertical Time	Funzione	-	Calcola il tempo necessario a spostarsi lungo l'asse verticale

Il primo flowchart *(Figura 7.15)* si occupa della gestione delle ShuttleMission. Anche in questo caso, come nel MaxiShuttle, il flusso si avvia nel blocco *enterMission* in cui sono inserite le missioni generate dall'agente Asrs che, successivamente, raggiungono il *wait* dove attendono di essere processate. Questo blocco rappresenta un primo elemento di differenza con il MaxiShuttle poiché le ShuttleMission sono gestite dalla *queue* del MaxiShuttle in modo da sincronizzare entrambe le missioni.

Nel *delay3* si richiama la funzione *f_CalculateVerticalTime* per il calcolo del tempo della missione, si aggiornano le informazioni relative alla missione attuale e, infine, lo Shuttle riceve il messaggio *seized* che genera un cambiamento di stato da *idle* a *moving* consentendo l'avvio della movimentazione.

I successivi *holdMaxiShuttle* e *holdShuttle* sono sbloccati quando il trasloelevatore si trova nello stato *arrived* ovvero all'arrivo, rispettivamente, del MaxiShuttle e dello Shuttle presso la destinazione della missione.

Infine, una volta raggiunto il *delay2* la missione giunge al termine, lo Shuttle passa dapprima allo stato *loadingUnloading* e, successivamente, torna ad essere inattivo. Inoltre, la conclusione di una missione sblocca il modulo *hold* consentendo l'avvio di una eventuale missione in coda.

Il diagramma di flusso termina con il blocco *exit2* in cui la missione è aggiunta alla collezione completedShuttleMission.

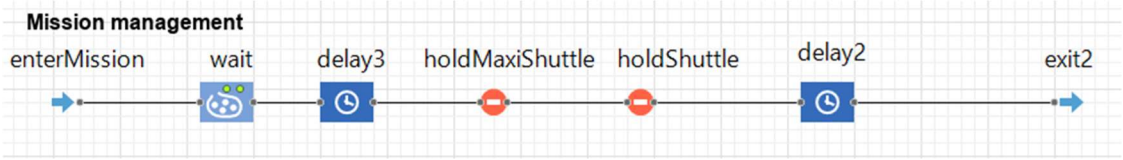


***Figura 7.15** – Flowchart Shuttle gestione ShuttleMission*

Il secondo flowchart *(Figura 7.16)* che caratterizza lo Shuttle riguarda, come detto, il caricamento delle Box sul trasloelevatore sia nel caso in cui si trovino sul conveyor di ingresso sia che si trovino stoccate nei vani.

Nel primo caso, la Box giunge nel flusso attraverso l'*enter* denominato *fromConveyor,* mentre nel blocco *delay4* subisce un ritardo pari al tempo di caricamento. Esso include l'ultimo tratto del conveyor ed è differente in base alla tipologia di cassetta.

Nel secondo caso, analogamente, la Box attraversa il ramo parallelo, passa per il blocco *enter* e raggiunge il *delay5* dove gli è assegnato un tempo di caricamento differente dal precedente perché relativo al prelievo dai vani; quest'ultimo dipende dal deep ovvero dalla profondità della Box all'interno del vano.

Il flusso si ricongiunge nel blocco *moveTo* in cui istantaneamente la Box è caricata su una delle due posizioni sul trasloelevatore in base al fronte di prelievo.

Successivamente, nel *delay1* si aggiornano i parametri e le variabili relative alle operazioni descritte e, inoltre, si interagisce con i diagrammi di flusso relativi alla gestione delle missioni sia del MaxiShuttle che dello Shuttle affinché la missione sia terminata.

Nuovamente il flusso si suddivide nel blocco *selectOutput* che invia la Box direttamente verso il flowchart dello StorageLocation se l'attuale locazione e quella di destinazione appartengono allo stesso fronte, altrimenti, la Box subisce un ritardo relativo allo spostamento sulla culla del trasloelevatore per poi raggiungere l'agente StorageLocation.

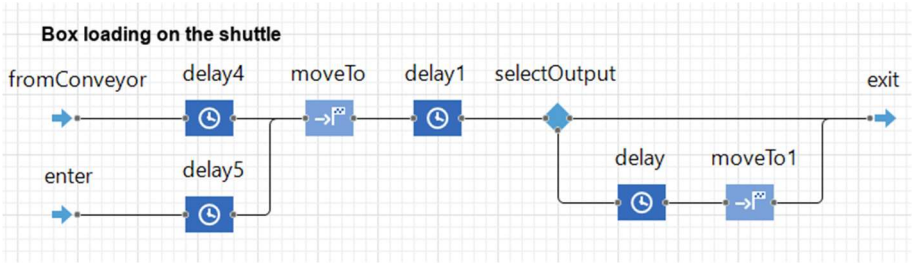


Figura 7.16 *– Flowchart Shuttle gestione Box*

7.1.8 MaxiShuttleMission/ShuttleMission

Gli agenti MaxiShuttleMission e ShuttleMission sono identici, tuttavia, sono stati rappresentati in due differenti agenti per semplicità di modellazione. L'unico elemento che li differenzia è il parametro tX per il primo e tZ per il secondo che identificano, rispettivamente, il valore del

tempo di percorrenza lungo l'asse x e l'asse z dei due agenti MaxiShuttle e Shuttle *(Tabella 7.14 – Attributi MaxiShuttleMission e ShuttleMission)*.

Entrambi gli agenti hanno lo scopo di determinare le missioni ovvero gli spostamenti operativi che il MaxiShuttle e lo Shuttle devono compiere per stoccare o prelevare delle Box. Ciò è possibile grazie ai parametri presenti nei due agenti che forniscono informazioni riguardo al tipo di missione, alle coordinate in cui il MaxiShuttle e lo Shuttle dovranno muoversi, alla box relativa alla missione e al vano in cui si trova o in cui essa dovrà essere stoccata in base alla tipologia di missione.

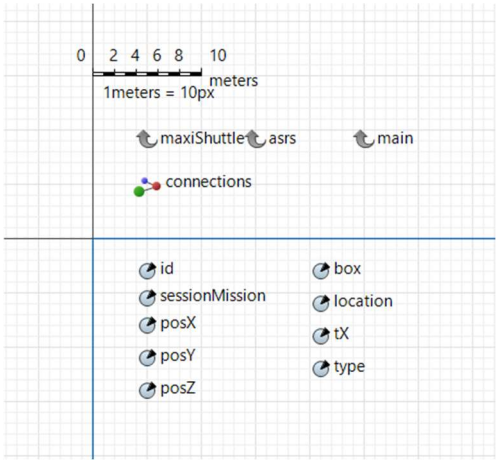


Figura 7.17 - *MaxiShuttleMission*

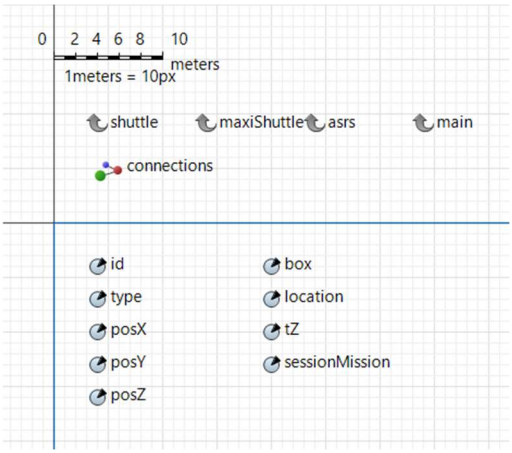


Figura 7.18 - *ShuttleMission*

Tabella 7.14 – *Attributi MaxiShuttleMission e ShuttleMission*

Nome	Elemento	Tipo	Descrizione
id	Parametro	double	Codice univoco per l'identificazione
sessionMission	Parametro	integer	Identifica il numero della Missione
type	Parametro	MissionType	Indica il tipo di Missione (storage o retrieval)
posX	Parametro	double	Indica la coordinata X in cui il trasloelevatore dovrà muoversi
posY	Parametro	double	Indica la coordinata Y in cui il trasloelevatore dovrà muoversi

Nome	Elemento	Tipo	Descrizione
posZ	Parametro	double	Indica la coordinata Z in cui il trasloelevatore dovrà muoversi
box	Parametro	Box	Identifica la Box soggetta alla Missione
location	Parametro	StorageLocation	Indica la locazione del vano in cui la Box è stoccata nel caso di Missione di retrieval, altrimenti indica il vano in cui sarà stoccata la Box nella Missione di storage
tX/tZ	Parametro	int	Tempo di percorrenza del MaxiShuttle lungo l'asse X/asse Z

7.1.9 StorageLocation

Per quel che riguarda la generazione dei vani del magazzino è stato possibile modellizzare una singola entità di StorageLocation, appartenente alla popolazione StorageLocations, caratterizzata dagli attributi presenti in *Figura 7.19* - StorageLocation e descritti in *Tabella 7.15*.

All'avvio della simulazione, sono estrapolate tutte le informazioni relative ai vani dal database *map_storage_locations* mediante linguaggio SQL e, di conseguenza, sono generate tutte le locazioni.

L'agente Storage Location è rappresentato graficamente da un rettangolo di colore grigio settato dalla funzione f_InitializeCompartment mentre i parametri servono a definirne la posizione e le dimensioni fisiche. Ogni vano in base alla propria lunghezza può essere suddiviso in 2 o 4 deep position, ovvero un diverso numero di posizioni. Tale compito è svolto dalla funzione f_CreateNodesDeepPositions che assegna alle variabili deep1-2-3-4 i corrispettivi Rectangular Node, vale a dire il modo in cui si definiscono gli spazi nel software Anylogic. In aggiunta, i deep generati sono inseriti nella collezione nodesDeepPositions.
In fase di esecuzione delle attività di stoccaggio, la funzione f_GetNode si occupa di attribuire alla Box la posizione nel vano; successivamente, le collezioni freeNodes e fullNodes sono aggiornate in modo tale da descrivere in ogni istante la situazione di ciascun vano.

Infine, sono presenti due differenti collezioni di Box, la prima denominata incomingBoxes che contiene le informazioni delle cassette in movimento verso i vani, mentre, la seconda, detta storedBoxes, rappresenta l'insieme delle box stoccate nel vano.

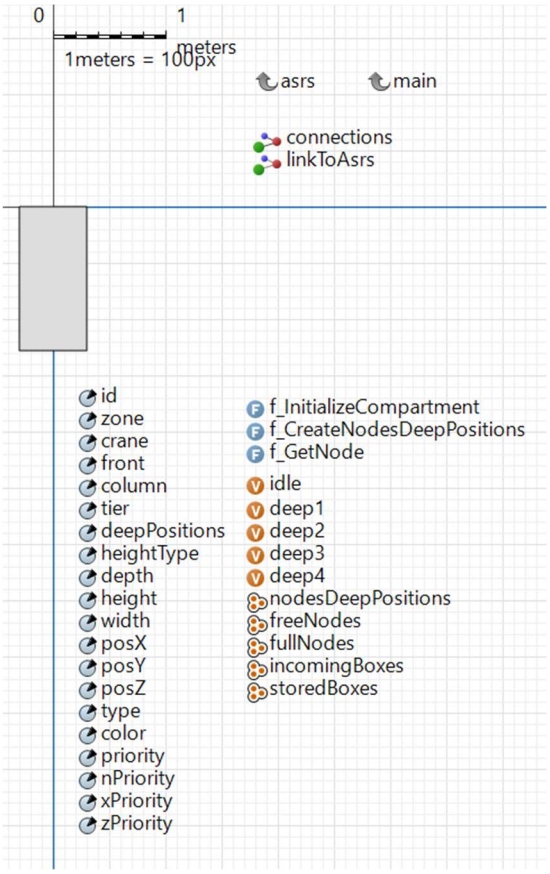


Figura 7.19 - *StorageLocation*

Tabella 7.15 – *Attributi StorageLocation*

Nome	Elemento	Tipo	Descrizione
id	Parametro	String	Codice univoco per l'identificazione
front	Parametro	Double	Indica il fronte in cui si trova il vano
column	Parametro	Double	Indica la colonna in cui si trova il vano
tier	Parametro	double	Indica il livello in cui si trova il vano

Nome	Elemento	Tipo	Descrizione
deepPositions	Parametro	int	Indica il numero di posizioni di stoccaggio nel vano
heightType	Parametro	int	Indica la tipologia del vano (1==basso o 2==alto)
depth	Parametro	double	Valore in mm della lunghezza
height	Parametro	double	Valore in mm dell'altezza
width	Parametro	double	Valore in mm della larghezza
posX	Parametro	double	Indica la posizione del vano lungo l'asse X
posY	Parametro	double	Indica la posizione del vano lungo l'asse Y
posZ	Parametro	double	Indica la posizione del vano lungo l'asse Z
type	Parametro	String	Identifica la tipologia del vano. N: vano di stoccaggio I: vano di ingresso U: vano di uscita UU: vano di uscita flow rack S: vano speciale
priority	Parametro	double	Indica il valore di priorità
nPriority	Parametro	double	Indica il valore di priorità normalizzato
idle	Variabile	boolean	Indica se il vano è coinvolto o meno in un'operazione
deep1-2-3-4	Variabile	RectangularNode	Indicano le posizioni all'interno di un vano
nodesDeep Positions	Collezione	ArrayList <RectangularNode>	Collezioni di posizioni all'interno del vano

Nome	Elemento	Tipo	Descrizione
freeNodes	Collezione	ArrayList <RectangularNode>	Collezione di nodi liberi del vano
fullNodes	Collezione	ArrayList <RectangularNode>	Collezione di nodi occupati del vano
incomingBoxes	Collezione	ArrayList<Box>	Lista di Box che stanno per essere stoccate
storedBoxes	Collezione	ArrayList<Box>	Lista di Box stoccate
f_Initialize Compartment	Funzione	-	Setta la posizione dell'agente Storage Location nello spazio
f_CreateNodes DeepPositions	Funzione	-	Per ogni vano assegna 2 o 4 deep positions in base alla grandezza del vano
f_GetNode	Funzione	-	Assegna alla Box la posizione nel vano

All'interno del vano è presente uno statechart che individua i tre possibili stati in cui ciascun vano si può trovare: empty, partiallyFull e full *(Figura 7.20)*. Il primo è lo stato iniziale e, come suggerisce il nome, indica che il vano è vuoto e, pertanto, pronto ad essere selezionato per ospitare una box. Il secondo stato è quello che rappresenta una condizione intermedia, ovvero il vano è parzialmente pieno e possono essere stoccate una o più unità di carico della stessa tipologia; infine, lo stato full indica che la locazione è completamente satura.

Affinché si realizzi un passaggio di stato è necessario che si verifichino alcune condizioni espresse sugli archi. In particolare, tali condizioni confrontano le dimensioni delle tre collezioni fullNodes, freeNodes e nodesDeepPositions che forniscono le informazioni relative alle profondità del vano occupate o meno.

Ad esempio, una volta che una cassetta è stata immagazzinata all'interno del vano, si esegue un controllo sulle condizioni di transizione e, in base al risultato, il vano aggiornerà il proprio stato in partiallyFull o full.

Inoltre, dalla *Figura 7.20*, si nota che lo stato partiallyFull può subire una transizione verso lo stato full e viceversa, e, da entrambi gli stati, il vano può tornare nello stato iniziale a patto che le giuste condizioni siano rispettate.

Di seguito sono espresse le 6 condizioni di transizione:

- da empty a partiallyFull: fullNodes.size() < nodesDeepPositions.size() && freeNodes.size() < nodesDeepPositions.size();

- da empty a full: fullNodes.size() == nodesDeepPositions.size();

- da partiallyFull a empty: freeNodes.size() == nodesDeepPositions.size();

- da partiallyFull a full: fullNodes.size() == nodesDeepPositions.size();

- da full a empty: freeNodes.size() == nodesDeepPositions.size();

- da full a partiallyFull: fullNodes.size() < nodesDeepPositions.size().

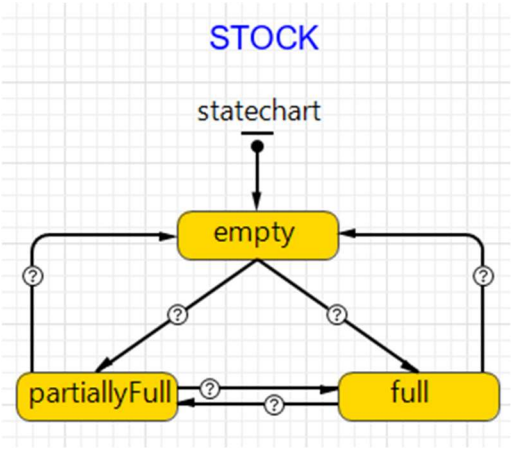


***Figura 7.20** – State chart StorageLocation*

Un ulteriore elemento presente nell'agente StorageLoacation è il flowchart che gestisce le cassette nei vani di locazione *(Figura 7.21)*. Il flusso inizia con un blocco *Enter* in cui le entità Box giungono dal diagramma di flusso relativo allo Shuttle. Successivamente, nel blocco *delay* a ciascuna Box è assegnato, tramite una query SQL applicata ad una tabella Excel, il tempo necessario al trasloelevatore a raggiungere la posizione nel vano, mentre, nel *moveTo2* avviene l'assegnazione di una posizione libera del vano oltre che l'effettivo rilascio della Box.

Il modulo *SelectOutput* svolge unicamente la funzione di indirizzare la Box verso il conveyor di uscita qualora il targetStorageLocation sia il vano di output; in particolare, quando l'agente Box transita nel blocco *toOutput* è trasferito nel flowchart dell'Asrs.

Il modulo *wait* rappresenta l'elemento più rilevante dell'intero flowchart; in esso la Box risiederà finché sarà richiamata attraverso un ordine di prelievo. Giunta la Box nel vano, sono aggiornate sia le informazioni relative ad essa, come la posizione attuale, sia le informazioni del vano e delle relative collezioni. Tali aggiornamenti generano, come visto nel paragrafo

precedente, cambiamenti di stato nello statechart della locazione corrispondente. Inoltre, in questo modulo i Task di Input giungono a conclusione e, mediante il richiamo di una funzione specifica, sono fatti avanzare nel flowchart del WMS.

Il modulo *Exit* alla fine del flusso permette alla cassetta di raggiungere il diagramma relativo al caricamento della Box nella culla del trasloelevatore, ovvero nella schermata relativa allo Shuttle, vista in precedenza.

In ultimo, è presente un ramo secondario del diagramma che permette di posizionare le cassette direttamente nei vani, senza che debbano effettivamente passare sui nastri trasportatori o essere movimentate dal trasloelevatore. Questo scenario si mette in atto solo se si vuole avere una situazione iniziale della simulazione differente dalla scaffalatura completamente vuota.

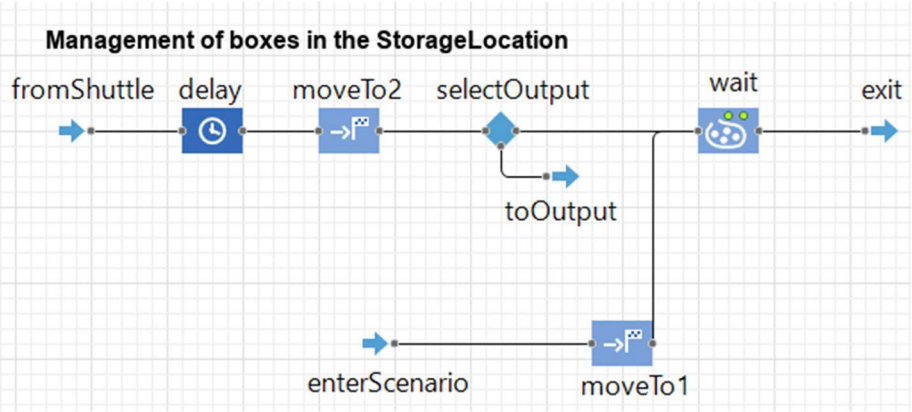


***Figura 7.21** – Flowchart StorageLocation*

8. Ottimizzazione

Dopo aver sviluppato il modello base su Anylogic in cui sono gestite le sessioni di task di input e output, la fase successiva è stata l'introduzione di un processo di ottimizzazione.Nel sistema reale, il trasloelevatore implementa un algoritmo di ottimizzazione affinchè il tempo per completare una sessione di task sia il minore possibile.

Per capire come il sistema reale effettua l'ottimizzazione, si è discusso con gli ingegneri dell'azienda produttrice del magazzino.

Attraverso questo confronto, è emerso che l'algoritmo di ottimizzazione utilizzato per gestire le sessioni, è l'algoritmo di DIJKSTRA, rinomato algoritmo di ottimizzazione usato per trovare il percorso più breve tra i nodi appartenenti ad un grafo. Questi nodi, come ci è stato detto, rappresentano lo stato del trasloelevatore, o meglio, lo stato del magazzino che varia ad ogni interazione che lo shuttle ha con le box inerenti alla sessione che si sta svolgendo.

Dopo ave raccolto queste informazioni, si è dovuto capire quali fossero le variabili necessarie e più adatte a rappresentare lo stato del magazzino ad ogni step del processo di stoccaggio.

Una volta stabilite le variabili si è svolta la fase di realizzazione del flowchart per la creazione del grafo.

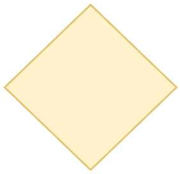
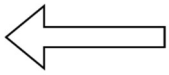
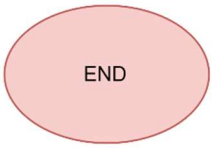
In un secondo momento, è stato sviluppato lo pseudo-codice relativo al flowchart implementato. Lo scopo dello psudo-codice è restituire in output una matrice origine-destinazione avente i nodi creati disposti sulle righe e colonne. Ad ogni intersezione della matrice è assegnato il peso dell'arco corrispondente al tempo impiegato per andare da un nodo ad un altro. Il tempo è ricavato dalla matrice dei tempi nella quale sono presenti i tempi che il traslo impiega per spostarsi da un vano all'altro.

8.1 Flowchart e pseudo-codice per la creazione del grafo

Come spiegato nel paragrafo precedente, per la costruzione del grafo si è, in primo luogo, creato il flusso logico, ovvero il flowchart, e, successivamente, si è creato lo pseudo-codice, il quale è possibile vederlo nella sua interezza in Appendice.

I blocchi usati nel diagramma di flusso sono descritti nella **Tabella 8.1.**

Tabella 8. 1 – Descrizione blocchi usati nel flowchart

Icona	Nome Blocco	Descrizione
	Source	Indica l'inizio del diagramma di flusso
	Data	Si passano le condizioni iniziali all'interno del flusso
	Process	Indica lo svolgimento di un processo
	Decision	Rappresenta un blocco in cui si verifica una condizione,la cui risposta è booleana
	Cycle	Usato come ciclo for
	Arrow	Usata per collegare blocchi distanti
	End	Indica la fine del diagramma di flusso

Per ogni Box oggetto di un Task della sessione corrente, si crea un collezione TBi contente la Box interessata e le cassette posizionate davanti ad essa, ordinate secondo la loro posizione all'interno dello storageLocation di riferimento. Nel caso in cui ci siano due o più task che

hanno per oggetto due Box differenti che si trovano nello stesso vano, avere collezioni separate creerebbe diversi problemi e aumenterebbe la complessità dell'algoritmo, pertanto, ogni TBi contenuta in una collezione TBj, è rimossa dal backlog.

Dopo aver definito il backlog, cioè le cassette che il trasloelevatore deve movimentare per completare la sessione, si crea il nodo di origine (come mostrato in **Figura 8.1**), che rappresenta la condizione di partenza del magazzino all'avvio della sessione.

Sono generate due liste, la prima, denominata *queue-node,* contiene tutti i nodi che sono creati e che devono essere ancora esplorati; la seconda, denominata *exploredNodes* che, come suggerisce il nome, contiene tutti i nodi che sono già stati esplorati.

Il nodo di origine è inserito nella lista *queue-node.*

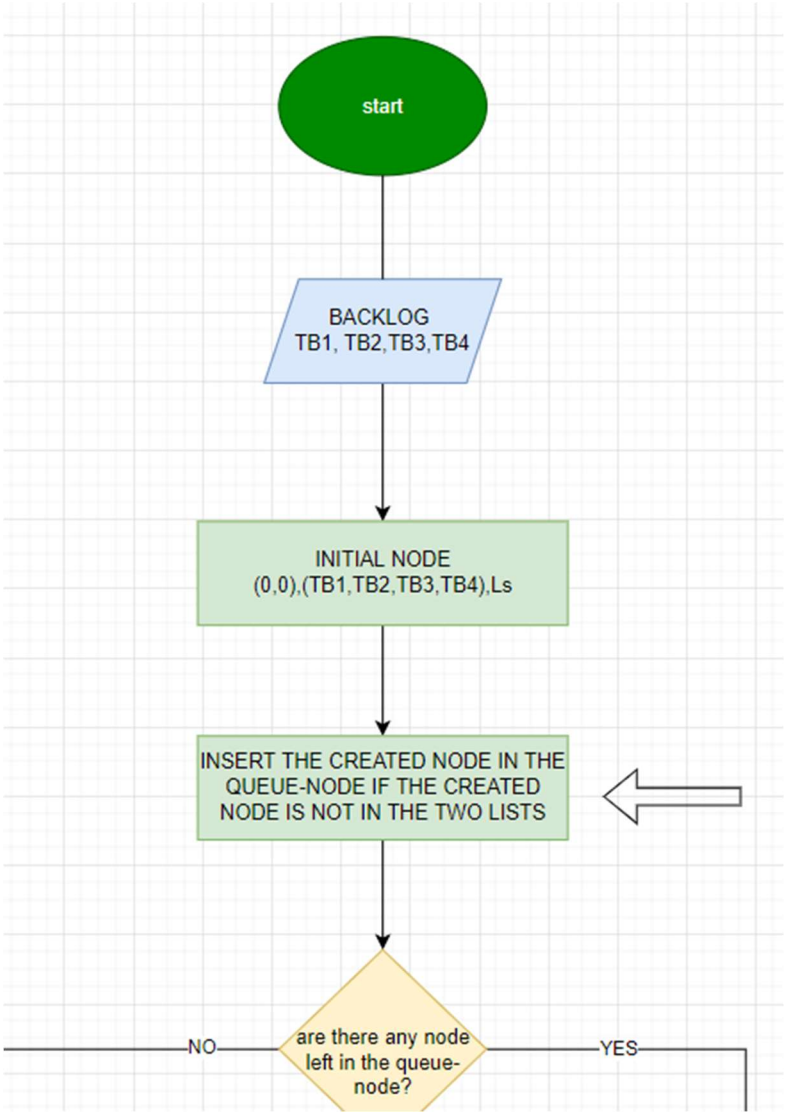


Figura 8. 1 – Rappresentazine fase iniziale del flowchart

Lo pseudo-codice che corrisponde a questa fase iniziale è:

```
f_Inizialize(){

        Creo lista di nodi ancora da esplorare : queueNodes [ArrayList<Node>]
        Creo lista di nodi esplorati :  exploredNodes  [ArrayList<Node>]

        f_defineBacklog() // funzione che definisce TB1,TB2,TB3,TB4 di backlog )

        L  =  storageLocation relativa all'ultima missione eseguita dello shuttle

        Node node  =  add_Nodes(idNode,null,null ,B,TB ,L) // funzione che crea un
        agente di tipo Node e assegna gli attributi iniziali.

        queueNode.add(node)   //si aggiunge il Nodo iniziale alla lista queueNode

}
```

Gli attributi che caratterizzano l'agente Node sono descritti in **Tabella 8.2 .**

Tabella 8. 2 – Parametri che caratterizzano i nodi del grafo

Nome	Elemento	Tipo	Descrizione
id	Parametro	integer	Codice univoco per l'identificazione
fatherNodeId	Parametro	ArrayList<int>	Lista di Id dei nodi padre
childNodeId	Parametro	ArrayList<int>	Lista di Id dei nodi figli
B[B1,B2]	Parametro	ArrayList<Box>	Lista di Box presenti sullo shuttle. B1 è relativa alla box posizionata verso il fronte 1 e B2 posizionata verso il fronte 2
T=[TBi,TBj..]	Parametro	ArrayList<TB>	Lista di TBi
TBi	Parametro	ArrayList<Box>	Lista di Box da movimentare per l'i-esimo task
L	Parametro	StorageLocation	Indica il vano in cui si trova lo shuttle

Come si vede in **Figura 8.2,** nel flowchart si procede verificando se ci sono ancora dei nodi da esplorare. Nel caso in cui non ci fossero, si crea il nodo finale ed è assegnato come nodo figlio a tutti i nodi che non ne hanno uno. Si tratta di un nodo fittizio che consente di chiudere il grafico con un unico nodo di fine, pertanto gli archi che lo collegano ai nodi padre hanno peso zero.

Dopo aver assegnato il nodo finale, il flowchart giunge alla fine, poichè sono stati creati ed esplorati tutti i nodi che possono essere percorsi per completare una sessione.

D'altra parte, nel caso in cui ci siano ancora dei nodi nella queue-node, se ne seleziona uno randomicamente affinchè possa essere esplorato.

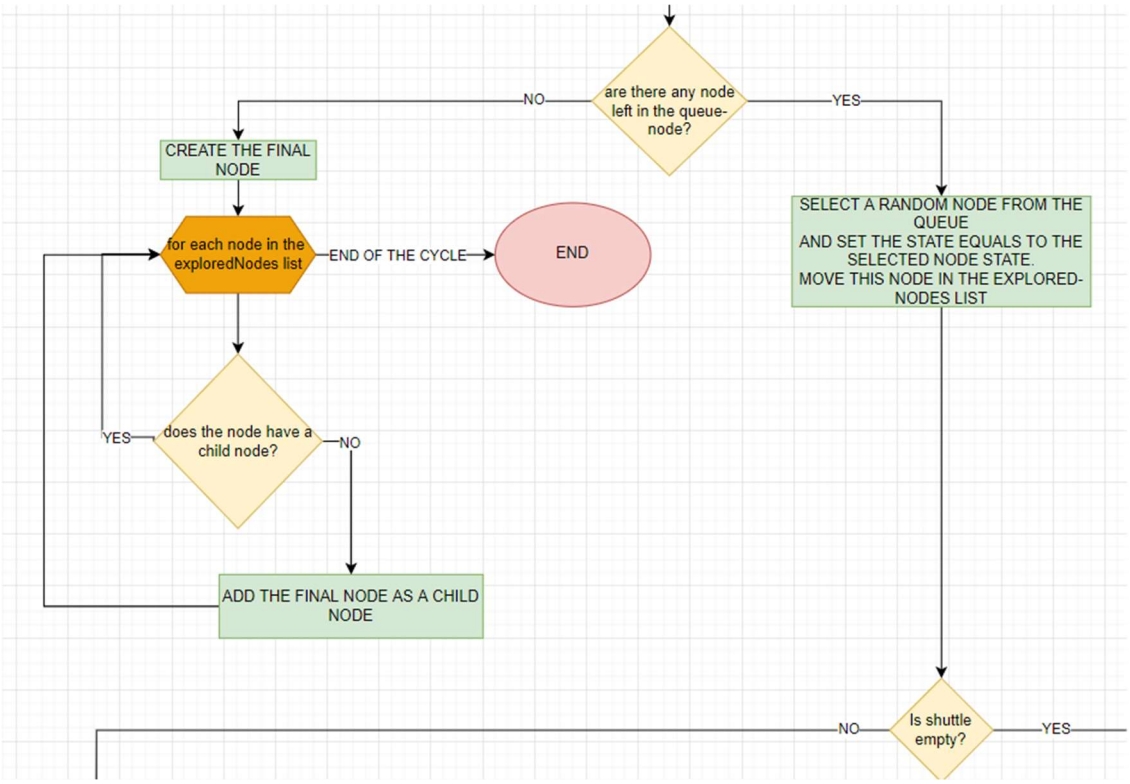


Figura 8. 2 – Creazione nodo finale

Dopo aver selezionato il nodo che si vuole esplorare, si controlla se ci siano delle box a bordo dello shuttle.

Lo Pseudo-codice relativo alla **Figura 8.2** :

```
// si verifica la condizione relativa al numero di cassette a bordo del trasloelevatore
       If ( condizione:  lo shuttle non ha nulla a bordo) {
              f_conditionShuttleEmpty() // funzione per gestire il caso zero cassette a
bordo
       }
       else{
       if (condizione:  lo shuttle ha una sola cassetta a bordo){
                     f_conditionOneBoxOnBoard() // funzione per gestire il caso una
              cassetta a bordo
              }
       else {
                     f_conditionTwoBoxOnBoard() // funzione per gestire il caso due
              cassette a bordo
```

```
        }
    }
```

Nel caso in cui non ci sia alcuna box sullo shuttle, come mostra il flowchart in **Figura 8.3**, si creano tanti nodi figli quanti sono i TBi. Per ogni TBi si controlla quante Box sono contenute e, di conseguenza, si caricano sullo shuttle una o due Box, creando un nodo con i parametri che dipendono dal vano di stoccaggio o di prelievo della Box. Il vettore T contente la lista di TBi è decrementato di uno nel momento in cui sono prelevate tutte le cassette presenti in quel TBi.

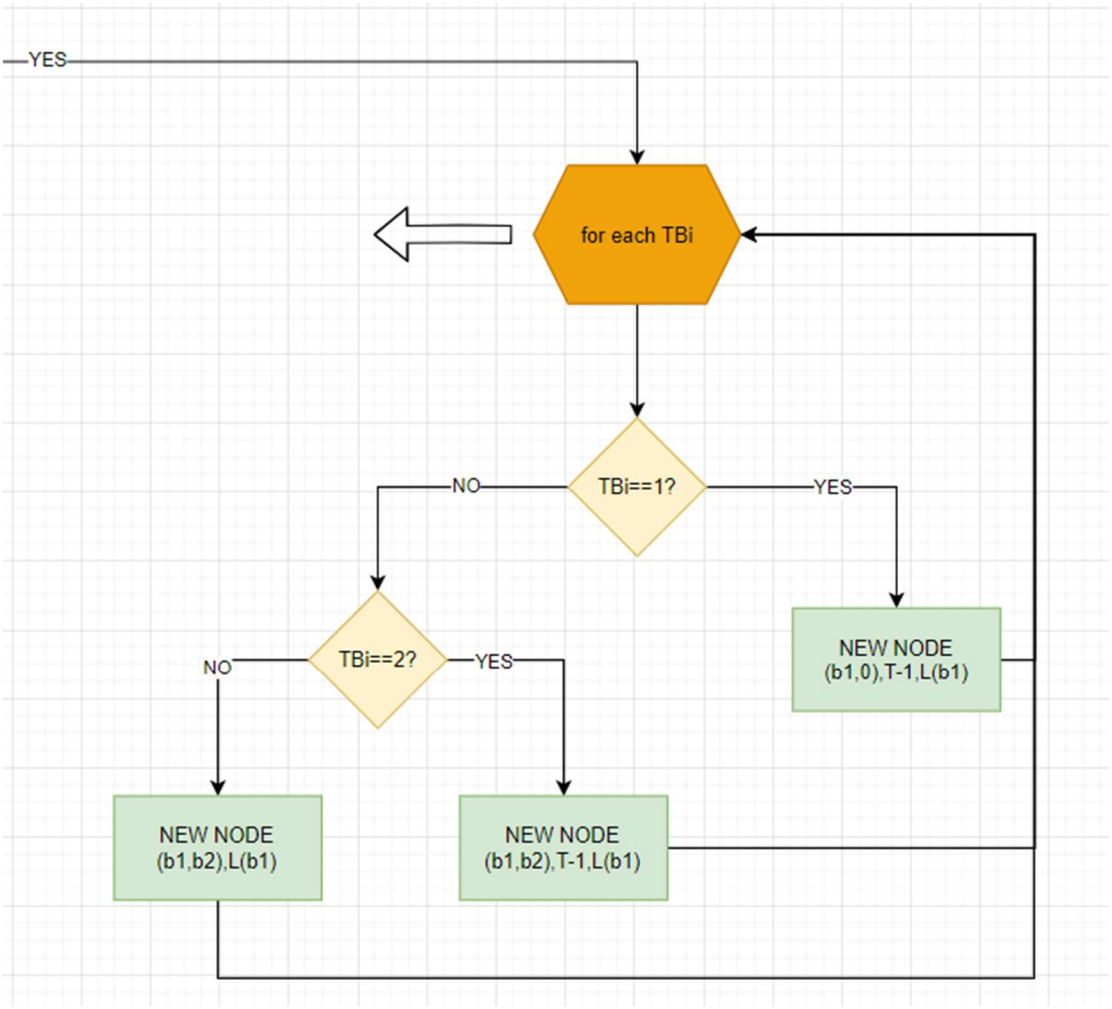


Figura 8. 3 – Creazione nodi nel caso in cui non vi sia alcuna cassetta sullo shuttle

E' proposto lo Pseudo-codice relativo alla **Figura 8.3:**

```
f_conditionShuttleEmpty() {
        For (ogni TBi, da TB1 a TB4) {
                ArrayList<Box> B;
                If (condizione: TBi contiene una sola cassetta) {
```

```
                        B = F_defineB(TBi.get(0),null); // in base al fronte su cui si
                        trova la cassetta, si assegna la box a B1 o B2.

                        TBi.remove(0); // si rimuove la box dal TBi dopo essere stata
                        prelevata
                }

                If (condizione: TBi contiene due cassette) {

                        B = F_defineB(TBi.get(TBi.Size-1), TBi.get(TBi.Size-2));

                        TBi.remove(TBi.Size-1);
                        TBi.remove(TBi.Size-1);

        }

        L= storageLocation in cui è situata la Box

        Node node = add_Nodes(idNode,fatherId,null,B, TB,L);

        F_assignChild(fatherId); // assegno il nodo creato come nodo figlio del nodo
        che si è esplorato

        queueNodes.add(node);  //aggiungo il nodo alla lista di nodi da esplorare


}


}
```

Nel caso in cui sul trasloelevatore ci sono due cassette, come mostrato in **Figura 8.4,** si controlla se le cassette sono appartenenti al backlog. Qualora entrambe lo siano, si crea un nodo caratterizzato da B=[null,B2] , che rappresenta lo scenario in cui è movimentata in uscita prima la cassetta posizionata verso il fronte 1. Il parametro L, relativo allo storage location, corrisponde al vano di uscita.

Se invece, nessuna delle due cassette è di backlog, si creano tre nodi fratelli relativi a differenti scenari. Il primo caso rappresenta il deposito di una delle due Box in un vano disponibile, viceversa, il secondo caso è speculare e riguarda il deposito dell'altra box. L'ultimo scenario considera lo stoccaggio di entrambe le cassette nello stesso vano in un'unica missione.

Qualora solo una delle due box è di backlog, si definiscono due nodi fratelli, come mostrato nel ciclo for, uno per compiere una movimentazione interna e uno per portare in uscita la cassetta richiesta.

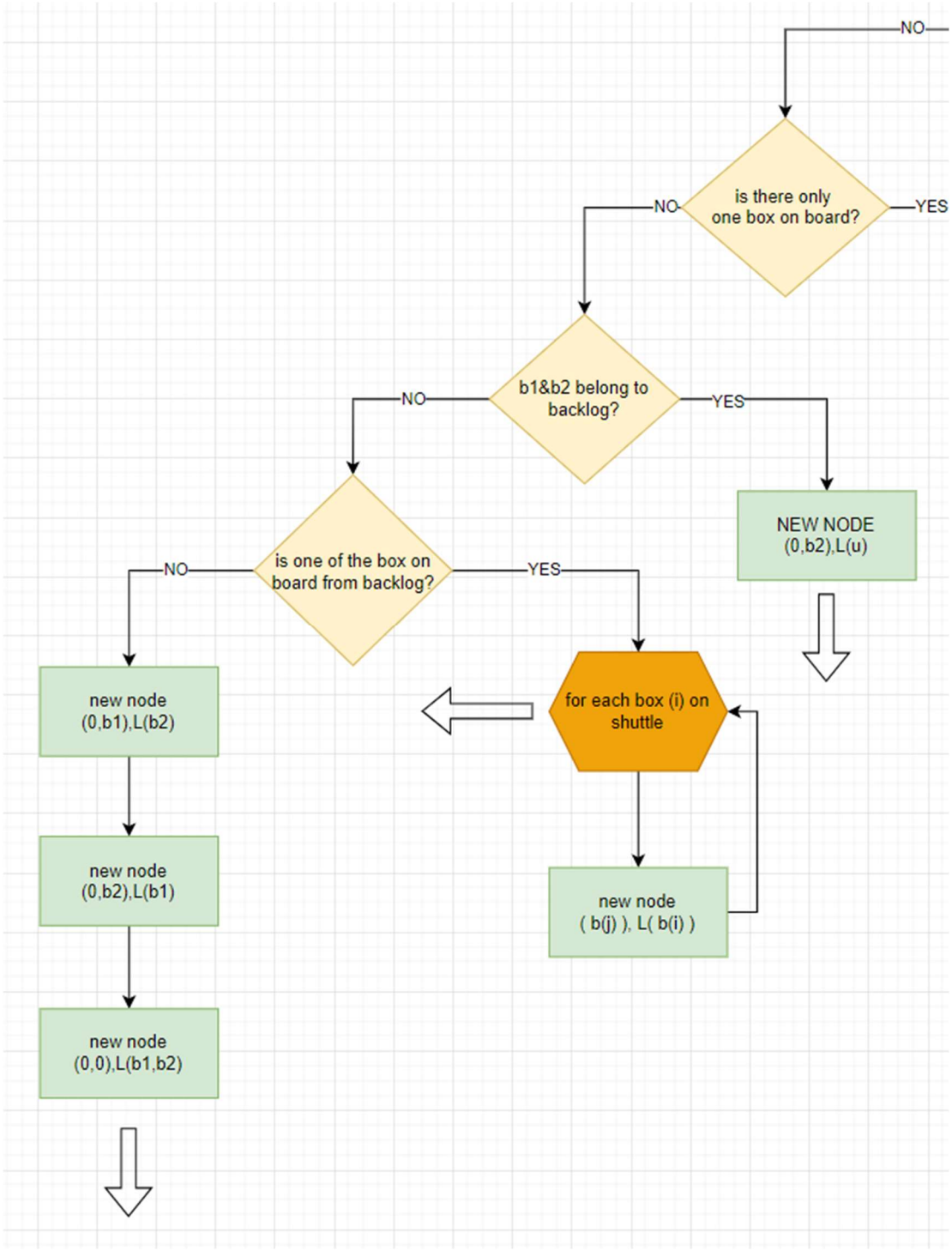


Figura 8. 4 - Creazione nodi nel caso in cui vi sono due cassetta sullo shuttle

```
f_conditionTwoBoxOnBoard(){
        // si esplora il caso in cui le cassette non siano entrambe di backlog
if(condizione: B1 or B2 è una delle Box oggetto di uno dei task della sessione ){
        For (ogni Box on board, da B1 a B2) {
                ArrayList<Box> B;

                B=[null,B2] or [B1,null] //dipende dalla Box che viene stoccata
```

```
                        L= storageLocation in cui la Box è stata stoccata

                        Node node = add_Nodes(idNode,fatherId,null,B, TB,L);

                        F_assignChild(fatherId); //assegno l'id del figlio, al nodo
                        padre

                        idNode=idNode+1; //incremento l'id poichè è univoco per ogni
                        nodo

                        queueNodes.add(node);  //aggiungo il nodo alla lista di nodi da
                        esplorare


                }
        }
        Else{
                //Creo tre nodi figli, e fratelli tra di loro
                        ArrayList<Box> B;

                        B=[null,B2] o B=[B1,null] o B=[null,null] //uno per ogni nodo


                        L= storageLocation in cui la Box è stata stoccata

                        Node node = add_Nodes(idNode,fatherId,null,B, TB,L);

                        F_assignChild(fatherId);

                        idNode=idNode+1;

                        queueNodes.add(node);  //aggiungo il nodo alla lista di nodi da
                        esplorare

        }
}
```

Nella condizione in cui sia presente una sola Box sulla culla del trasloelevatore e, contestualmente non vi siano altre cassette in backlog da movimentare, allora lo shuttle deposita la Box in uscita, se di backlog, altrimenti in un vano disponibile. Vedi **Figura 8.5.**

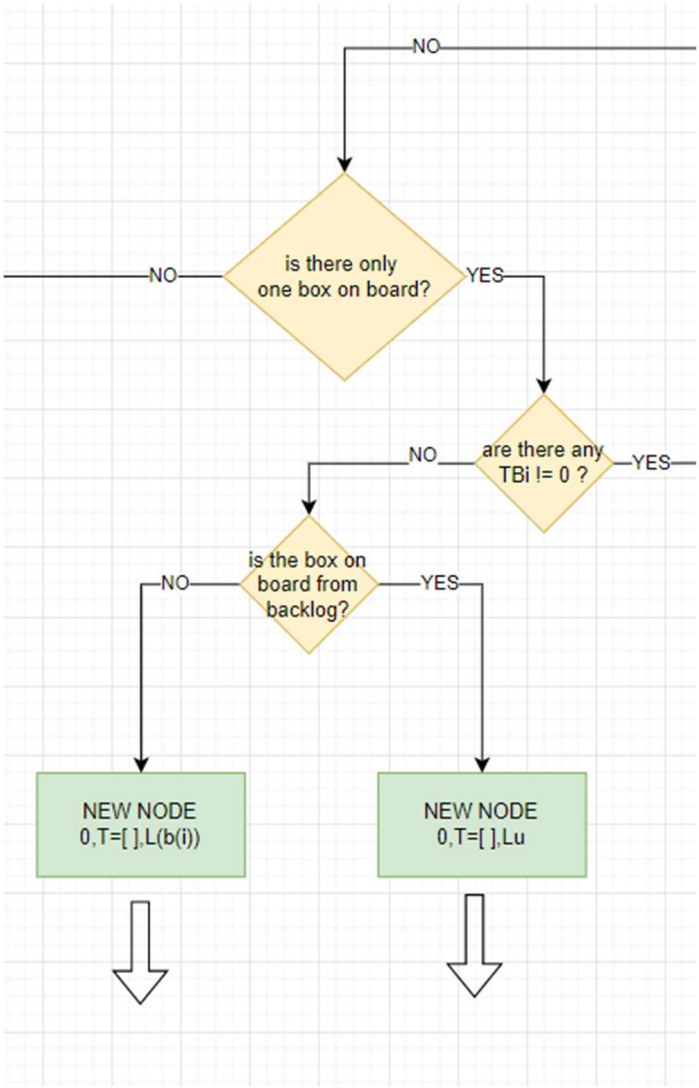


Figura 8. 5 - Creazione nodi nel caso in cui vi è una cassetta sullo shuttle e non vi sono altre cassette in backlog

Infine, se la domanda alla risposta precedente è positiva, cioè se ci sono ancora delle Box in backlog, si creano due nodi fratelli come si può vedere in **Figura 8.6** : un nodo rappresenta lo scenario in cui si deposita la cassetta a bordo e l'altro, il caso in cui si decide di prelevare a bordo dello shuttle un'altra casetta, a condizione che sia dello stesso tipo di quella già presente.

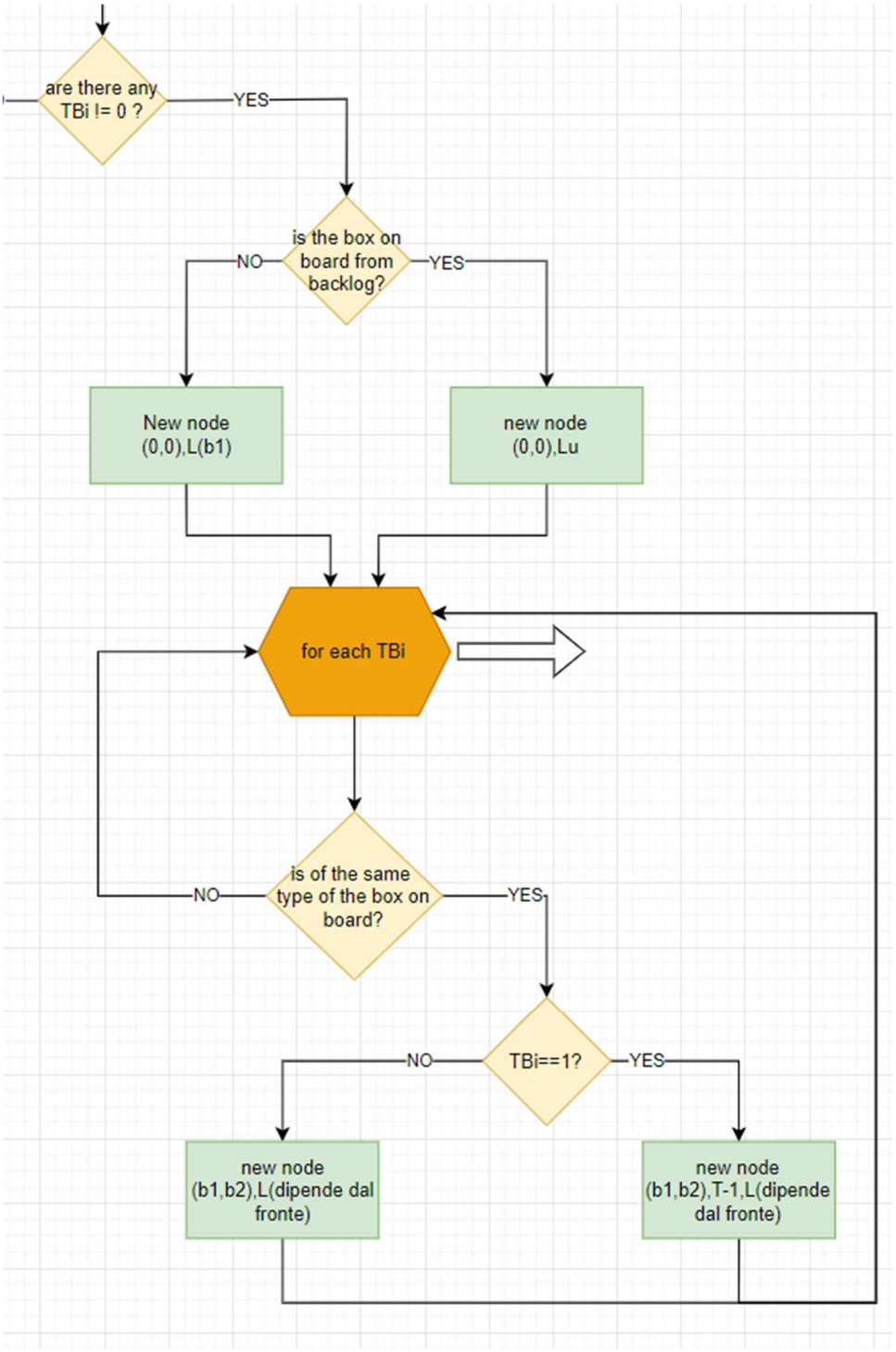


Figura 8. 6 - Creazione nodi nel caso in cui vi è una cassetta sullo shuttle e vi sono altre cassette in backlog

Lo pseudo-codice relativa a qusta parte del flusso, cioè se vi è solo una box a bordo e vi sono ancora altre box nel backlog da movimentare:

```
f_conditionOneBoxOnBoard(){
        // caso in cui vi siano box da movimentare nel backlog
                    B=[null,null]
                    L= storageLocation in cui la Box è stata stoccata
                    Node node = add_Nodes(idNode,fatherId,null,B, TB,L);
                    F_assignChild(fatherId);
                    idNode=idNode+1;
                    queueNodes.add(node);
                    For ( each TBi ){
                     If ( condizione: Il Box type di Tbi is equal to the that of
                    the box on board ){
                                TBnew=TB;  //perchè altrimenti il TB ( array di
                                TBi) che inserisco nel nodo è diverso ad ogni
                                for
                           //A seconda del fronte su cui si trova la Box che si
                           sta prelevando, cambieranno B1 e B2
              B=[B1, Tbnew.get(i).get(TBi.Size-1)] ||[TBnew.get(i).get(TBi.Size-
       1),B2]
                                TBnew.get(i).remove(TBi.Size-1);
                         L= storageLocation in cui è situata la Box
                                Node node = add_Nodes(idNode,fatherId,null,B,
                                TBnew,L);
                                F_assignLastChild(fatherId);
                                idNode=idNode+1;
                                queueNodes.add(node);
                           }
                    }
       }
}
```

8.2 Conclusione

E’ importante tenere in considerazione che l’algoritmo di creazione del grafo richiede in input lo stato iniziale del magazzino all’avvio della sessione, poichè il grafo costruito è dinamico e differente ad ogni sessione.

Il grafo, rappresentato dalla matrice origine-destinazione, infine è dato in input all’algoritmo di Dijkstra, il quale restitiusce la lista ordinata di nodi del percorso più breve che rappresenta l’ordine con cui lo shuttle deve eseguire le varie missioni per completare la sessione nel minore tempo possibile.

9. Discussioni dei risultati e conclusioni

La ricerca svolta ha portato ai risultati prefissati, ovvero la costruzione di un modello di simulazione del magazzino, la realizzazione di un algoritmo di ottimizzazione per il processo di prelievo e lo sviluppo di un modello preliminare per un Digital Twin. L'impiego di Anylogic come ambiente in cui sviluppare il modello virtuale si è rivelata una buona scelta, tuttavia, è impossibile negare la presenza di difficoltà iniziali legate alla complessità del software. Anylogic è un software di simulazione che integra l'agent-based simulation, la discrete event simulation e la programmazione object-oriented. Tuttavia, una volta superate le difficoltà iniziali legate alla simulazione multi-metodo, Anylogic ha rivelato tutto il suo potenziale. Come si può facilmente dedurre, l'elevata complessità riscontrata si traduce in un'elevata flessibilità nel processo di simulazione. Inoltre, la possibilità di costruire i propri agenti e oggetti, come nel caso del MaxiShuttle e Shuttle, conferisce un valore aggiunto a questo software.

I principali aspetti che non sono stati implementati nel modello di simulazione, nonostante siano presenti nel sistema reale, riguardano la stazione di picking e quella di kitting e i relativi processi ad esse legate. La scelta di escludere dal lavoro di tesi questi aspetti risiede nel tentativo di semplificare la complessità del magazzino presente in laboratorio. Inoltre, per la stessa ragione, il modello non include la possibilità di movimentare contemporaneamente due cassette sulla culla del trasloelevatore durante il processo di prelievo. Pertanto, data la mancanza di questo aspetto fondamentale per il processo di gestione degli ordini di prelievo, non è stato possibile realizzare la validazione di questa componente e, di conseguenza, dell'intero modello. Spunti di ricerca interessanti riguardano certamente le criticità elencate. In aggiunta, può essere notevole lo sviluppo e la realizzazione di un vero e proprio Digital Twin per studiare l'ottimizzazione dei processi e intervenire autonomamente per integrare i cambiamenti necessari.

In conclusione, è possibile affermare che, nonostante tutte le complicazioni e le problematiche riscontrate, i risultati ottenuti sono positivi e rappresentano un buon lavoro preliminare per lo sviluppo di un Digital Twin.

Sviluppi futuri del lavoro di tesi potrebbero interessare l'applicazione del modello di simulazione a contesti reali come, ad esempio, l'implementazione di un magazzino automatico a supporto delle linee produttive che utilizzano componentistica di piccola/media dimensione. Inoltre, dal momento che il modello è realizzato in forma parametrica risulta un ottimo strumento per analizzare la progettazione di un nuovo magazzino o per valutare differenti scenari di utilizzo di uno esistente. Le leve su cui si può agire riguardano le dimensioni del magazzino, il numero e la tipologia delle cassette, dei vani, dei conveyor in input e output e le stazioni di lavoro, oltre che, tutte le variabili e i parametri presenti nell'attuale configurazione.

BIBLIOGRAFIA

Altiok, T., & Melamed, B. (2010). Simulation modeling and analysis with Arena. Elsevier.

Ashayeri, J., & Gelders, L. F. (1985). Warehouse design optimization. *European Journal of Operational Research*, *21*(3), 285-294.

Baker, P., & Canessa, M. (2009). Warehouse design: A structured approach. *European journal of operational research*, *193*(2), 425-436.

Bergs, T., Gierlings, S., Auerbach, T., Klink, A., Schraknepper, D., & Augspurger, T. (2021). The Concept of Digital Twin and Digital Shadow in Manufacturing. *Procedia CIRP*, *101*, 81-84.

Chen, J., Xu, S., Liu, K., Yao, S., Luo, X., & Wu, H. (2022). Intelligent Transportation Logistics Optimal Warehouse Location Method Based on Internet of Things and Blockchain Technology. *Sensors*, *22*(4), 1544.

Chen, X., Ong, Y. S., Tan, P. S., Zhang, N., & Li, Z. (2013, October). Agent-based modeling and simulation for supply chain risk management-a survey of the state-of-the-art. In 2013 IEEE International Conference on Systems, Man, and Cybernetics (pp. 1294-1299). IEEE.

Choi, T. Y., Dooley, K. J., & Rungtusanatham, M. (2001). Supply networks and complex adaptive systems: control versus emergence. *Journal of operations management*, *19*(3), 351-366.

Fahl, S. K. (2017). Benefits of Discrete Event Simulation in Modeling Mining Processes.

Frazelle, E. H. (2016). World-class warehousing and material handling. McGraw-Hill Education.

Ghiani, G., Laporte, G., & Musmanno, R. (2004). Introduction to logistics systems planning and control. John Wiley & Sons.

Gong, Y. (2009). Stochastic modelling and analysis of warehouse operations (No. EPS-2009-180-LIS).

Gong, Y., & De Koster, R. (2008). A polling-based dynamic order picking system for online retailers. IIE transactions, 40(11), 1070-1082.

Hamdy, W., Al-Awamry, A., & Mostafa, N. (2022). Warehousing 4.0: A proposed system of using node-red for applying internet of things in warehousing. *Sustainable Futures*, *4*, 100069.

Hearnshaw, E. J., & Wilson, M. M. (2013). A complex network approach to supply chain network theory. *International Journal of Operations & Production Management*.

Kamalli, A. (2019). Smart warehouse vs. traditional warehouse. *Int. J.*

Kosanić, N. Ž., Milojević, G. Z., & Zrnić, N. Đ. (2018). A survey of literature on shuttle based storage and retrieval systems. *FME Transactions*, *46*(3), 400-409.

Kulturel, S., Ozdemirel, N. U. R., Sepil, C., & Bozkurt, Z. (1999). Experimental investigation of shared storage assignment policies in automated storage/retrieval systems. *IIE transactions*, *31*(8), 739-749.

Lehmann, T., & Hußmann, J. (2021). Travel time model for multi-deep automated storage and retrieval system with a homogeneous allocation structure.

Liu, Z., Wang, Y., Jin, M., Wu, H., & Dong, W. (2021). Energy consumption model for shuttle-based Storage and Retrieval Systems. *Journal of Cleaner Production*, *282*, 124480.

Loper, M. L. (Ed.). (2015). Modeling and simulation in the systems engineering life cycle: core concepts and accompanying lectures. Springer.

Maka, A., Cupek, R., & Wierzchanowski, M. (2011, March). Agent-based modeling for warehouse logistics systems. In 2011 UkSim 13th International Conference on Computer Modelling and Simulation (pp. 151-155). IEEE.

Mason, R.J. (2009). Collaborative logistics triads in supply chain management.
Ramaa, A., Subramanya, K. N., & Rangaswamy, T. M. (2012). Impact of warehouse management system in a supply chain. International Journal of Computer Applications, 54(1).

Sargent, R. G. (1979). Validation of simulation models. Institute of Electrical and Electronics Engineers (IEEE).

Sepasgozar, S. M. (2021). Differentiating digital twin from digital shadow: Elucidating a paradigm shift to expedite a smart, sustainable built environment. Buildings, 11(4), 151.

Sepasgozar, S. M. (2021). Differentiating digital twin from digital shadow: Elucidating a paradigm shift to expedite a smart, sustainable built environment. Buildings, 11(4), 151.

Simulation Modeling And Analysis with Arena.

Tompkins, J. A., White, J. A., Bozer, Y. A., & Tanchoco, J. M. A. (2010). Facilities planning. John Wiley & Sons.

Park, G. J. (2007). Design of experiments. Analytic methods for design practice, 309-391.

Uy, M., & Telford, J. K. (2009, March). Optimization by design of experiment techniques. In 2009 IEEE Aerospace conference (pp. 1-10). IEEE.

Agalianos, K., Ponis, S. T., Aretoulaki, E., Plakas, G., & Efthymiou, O. (2020). Discrete event simulation and digital twins: review and challenges for logistics. Procedia Manufacturing, 51, 1636-1641.

Liong, C. Y., & Loo, C. S. (2009). A simulation study of warehouse loading and unloading systems using Arena. Journal of Quality Measurement and Analysis, 5(2), 45-56.

Gagliardi, J. P., Renaud, J., & Ruiz, A. (2007, December). A simulation model to improve warehouse operations. In 2007 Winter Simulation Conference (pp. 2012-2018). IEEE.

Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numerische mathematik*, *1*(1), 269-271.

Mehlhorn, K., & Sanders, P. (2008). Shortest paths. *Algorithms and Data Structures: The Basic Toolbox*, 196-206.

SITOGRAFIA

AS/RS Systems: What is an Automated Storage and Retrieval System? (6river.com) && Automated Storage & Retrieval System (AS/RS) Types & Uses (conveyco.com) && Automated Warehouse Storage Systems - Types, Pros & Cons | #HowToRobot

Bartholdi III JJ, Hackman ST (2006), Warehouse and distribution science. https://www.warehouse-science.com

Diversi diagrammi UML - Scopo e utilizzo (edrawsoft.com)

https://it.blog.kardex-remstar.com/miti-da-sfatare-logistica/magazzino-automatico-orizzontale-carosello-tu-quanto-ne-sai

https://www.berkshiregrey.com/asrs-picking/

https://www.ionos.it/digitalguide/online-marketing/vendere-online/supply-chain-management/

https://www.jungheinrich.ch/it/systeme/automatizzate-il-vostro-successo-logistico/magazzino-per-pallet-automatico/trasloelevatore-per-pallet-468610

https://www.mobile-industrial-robots.com/it/solutions/robots/mir100/

https://www.scaffsystem.it/blog/credito-dimposta-e-magazzini-automatizzati/

https://www.viastore.com/systems/en-us/warehouse-and-material-flow-solutions/automated-mini-load-storage-0

https://www.sciencedirect.com/topics/computer-science/dijkstra-algorithms

https://www.sciencedirect.com/topics/computer-science/directed-graphs

https://mathworld.wolfram.com/WeightedGraph.html

APPENDICE

Pseuo-codice creazione grafo

Il seguente pseudo-codice si basa sul linguaggio di programmzione java.

```
Int idNode=0; ( per assegnare un id univoco ad ogni nodo creato , variabile esterna )

Agent   Node {

Id [int]

nodoPadreId  [ArrayList<int>]

NodoFiglioId  [ArrayList<int>]

B = [B1,B2] dove B1 [Box] verso fronte 1 e B2 [Box] verso fronte 2

TB=[TB1,TB2,TB3,TB4] dove:

TB1 [C

TB2 [ArrayList<Box>]

TB3, [ArrayList<Box>]

TB4 [ArrayList<Box>]

L [StorageLocation]

}


f_defineBacklog{

Per ogni Box oggetto di un Task della sessione corrente, si crea un array TBi contente le
Box ordinate secondo la loro posizione all'interno dello storageLocation di riferimento.
Per ogni TBi, se è contenuto  un TBj , e quindi ci sono due task che hanno oggetto due Box
che si trovano sullo stesso vano, allora TBi viene rimosso.

}

F_assignChild(id){  collego il nodo figlio al nodo padre

       For ( ogni nodo i nella lista exploredNodes)

              {

              If ( exploredNodes.get(i).id == id){

                     exploredNodes.get(i).nodoFiglioId.add(idNode);

              }

       }

}
```

```
F_defineB(b1,b2){  dove b1 si trova dvanti a b2, e se c'è una sol Box, b2=null
        ArrayList<Box> B;
        if (front in cui si trova la Box == 1){
                B=[b1,b2];
        }
        else{
                B=[b2,b1];
        }
        Return B;
}
```

Fase 1.

```
        f_Inizialize(){
                Creo lista di nodi ancora da esplorare : queueNodes [ArrayList<Node>]
                Creo lista di nodi esplorati :  exploredNodes  [ArrayList<Node>]

                f_defineBacklog() [function] //definisce TB1,TB2,TB3,TB4 dello stato iniziale

                L = storageLocation relativa all'ultima missione eseguita dello shuttle

                Node node = add_Nodes(idNode,null,null ,B,TB ,L) [function] //crea una
                istanza dell'agente Node e assegna un valore a ciascuna variabile
                idNode = idNode+1;

                queueNode.add(node)   // aggiungo il Nodo iniziale alla lista queueNode
        }
```

Fase 2.

```
Node currentNode=null; // Inizializzo la variabile a null

If (nella lista queueNode sono presenti uno o più nodi da esplorare){

        //Node currentNode = a random node from queueNode

        // Assegno alla variabile currentNode [node] un nodo random presente nell list
        queueNode

        // Dopo aver definito il nodo da esplorare

        if ( nelle liste queueNode e exploredNodes NON esiste un nodo che ha gli stessi
        attributi del currentNode tranne che per il nodoPadre e il nodoFiglio) {

                sposto il nodo selezionato dalla queueNode alla list exploredNodes

                exploredNodes.add(currentNode);

                queueNode.remove(currentNode);

                fatherId = currentNode.Id;

        }

        else {

                Fine ( L'ultimo nodo esplorato è il nodo di fine, e non ha nodi figli )

                Si ricomincia ad esplorare i nodi in coda, partendo dal punto 2.

        }

}

else {

        //ASSIGN THE FINAL NODE

        Node finalNode = add_Nodes(idNode,fatherId,null,B=[], TB=[],Lf);

        For ( each node in the exploredNodes list){

        If (the node(i) doesn't have a child node){

                fatherId= node(i).idNode

                        F_assignChild(fatherId);

                }

        }

}

        Controllo se lo shuttle non abbia nulla a bordo

        If ( B1==null && B2==null ) {

                f_conditionShuttleEmpty()

        }

        else{

        if (B1!=null || B2!=null) f_conditionOneBoxOnBoard()

        else f_conditionTwoBoxOnBoard()

        }

}
```

```
f_conditionShuttleEmpty()

{

              For (ogni TBi, da TB1 a TB4) {
                     ArrayList<Box> B;

                     If (TBi.Size == 1) {

                            B = F_defineB(TBi.get(0),null);

                            TBi.remove(0);
                     }

                     If (TBi.Size == 2) {

                            B = F_defineB(TBi.get(TBi.Size-1), TBi.get(TBi.Size-
                     2));

                            TBi.remove(TBi.Size-1);
                            TBi.remove(TBi.Size-1);

              }

              L= storageLocation in cui è situata la Box

              Node node = add_Nodes(idNode,fatherId,null,B, TB,L);

              F_assignChild(fatherId);

              idNode=idNode+1;

              queueNodes.add(node);  aggiungo il nodo alla lista di nodi da
              esplorare


       }

}



f_conditionTwoBoxOnBoard(){

if( B1 or B2 sono entrambe delle Box oggetto dei task della sessione ){

              ArrayList<Box> B;

              B=[null,B2];

              L= storageLocation in cui la Box è stata stoccata ( outputLocation )

              Node node = add_Nodes (idNode,fatherId,null,B, TB,L);

              F_assignChild(fatherId);

              idNode=idNode+1;

              queueNodes.add(node);  aggiungo il nodo alla lista di nodi da
              esplorare

       }

if( B1 or B2 è una delle Box oggetto di uno dei task della sessione ){

       For (ogni Box on board, da B1 a B2) {
                     ArrayList<Box> B;

              B=[null,B2] or [B1,null] a seconda della Box che viene stoccata
```

```
                        L= storageLocation in cui la Box è stata stoccata

                        Node node = add_Nodes(idNode,fatherId,null,B, TB,L);

                        F_assignChild(fatherId);

                        idNode=idNode+1;

                        queueNodes.add(node);  aggiungo il nodo alla lista di nodi da
                        esplorare


                }
        }
        Else{
                Creo tre nodi figli, e fratelli tra di loro

                        ArrayList<Box> B;

                B=[null,B2]

                        L= storageLocation in cui la Box è stata stoccata

                        Node node = add_Nodes(idNode,fatherId,null,B, TB,L);

                        F_assignChild(fatherId);

                        idNode=idNode+1;

                        queueNodes.add(node);  aggiungo il nodo alla lista di nodi da
                        esplorare

                B=[B1,null];

                        L= storageLocation in cui la Box è stata stoccata

                        Node node = add_Nodes(idNode,fatherId,null,B, TB,L);

                        F_assignChild(fatherId);

                        idNode=idNode+1;

                        queueNodes.add(node);  aggiungo il nodo alla lista di nodi da
                        esplorare

                        B=[null,null]

                        L= storageLocation in cui la Box è stata stoccata

                        Node node = add_Nodes(idNode,fatherId,null,B, TB,L);

                        F_assignChild(fatherId);

                        idNode=idNode+1;

                        queueNodes.add(node);  aggiungo il nodo alla lista di nodi da
                        esplorare

        }
}


f_conditionOneBoxOnBoard(){

Check if there are still tasks to do

taskToDo= false;

for ( each TBi ){
```

```
        if (TBi.length >0 ) taskToDo=true; }

        If (taskToDo== false) {

               Assign to the current node explored, the final node as his node child

                      B=[null,null]

                      L= storageLocation in cui la Box è stata stoccata

                      Node node = add_Nodes(idNode,fatherId,null,B, T= [ ],Lui);

                      F_assignChild(fatherId);

                      idNode=idNode+1;

                      queueNodes.add(lastNode);  aggiungo il nodo alla lista di nodi
                      da esplorare

        }

        If (taskToDo== true) {

                      B=[null,null]

                      L= storageLocation in cui la Box è stata stoccata

                      Node node = add_Nodes(idNode,fatherId,null,B, TB,L);

                      F_assignChild(fatherId);

                      idNode=idNode+1;

                      queueNodes.add(node);  aggiungo il nodo alla lista di nodi da
                      esplorare

                      For ( each TBi ){

                       If ( Il Box type di Tbi is equal to the that of the box on
                      board ){

                                    TBnew=TB;  perchè altrimenti il TB ( array di
                                    TBi) che inserisco nel nodo è diverso ad ogni
                                    for

                             A seconda del fronte su cui si trova la Box che si sta
                             prelevando, cambieranno B1 e B2

               B=[B1, Tbnew.get(i).get(TBi.Size-1)]||[TBnew.get(i).get(TBi.Size1),B2]

                                    TBnew.get(i).remove(TBi.Size-1);

                              L= storageLocation in cui è situata la Box

                                    Node node = add_Nodes(idNode,fatherId,null,B,
                                    TBnew,L);

                                    F_assignLastChild(fatherId);

                                    idNode=idNode+1;

                                    queueNodes.add(node);  aggiungo il nodo alla
                                    lista di nodi da esplorare

                             }

                      }

        }

}
```