POLITECNICO DI TORINO

Corso di Laurea Magistrale

in Ingegneria Meccanica

Tesi di laurea Magistrale

Autonomous Vehicle Storage and Retrieval Systems:
Studio dei fattori d'incidenza sui KPI attraverso
un modello ad eventi discreti

Relatore:

Prof. Franco Lombardi

Co-Relatori:

Dott. Emiliano Traini

Candidato:

Alfredo De Filippis

Anno Accademico 2018/2019

≪ Io voglio il sole dentro me,
io cerco il sole dentro di te ≫

Pino Daniele

Sommario

Introduzione

La scena globale della produzione industriale, negli ultimi anni, ha subito importanti cambiamenti, per sopravvivere ad una riduzione drastica della domanda e la sempre maggiore personalizzazione degli ordini ricevuti. Infatti, oggi, la gestione di un impianto ha sempre più importanza, con la necessità di ridurre tempi e costi di spedizione e l'esigenza di una profonda ristrutturazione del settore logistico [1].
In particolare, il servizio è valutato in termini di:

- Disponibilità del prodotto, con scorte sempre sufficienti in magazzino;
- Tempestività delle consegne al cliente, evitando ritardi che ledono l'immagine dell'aziende e possono prevedere delle penali;
- Affidabilità del servizio, con regolarità del tempo di consegna, conformità ed integrità dei prodotti;
- Flessibilità del servizio, capacità di adattare il sistema logistico alle richieste del cliente.

L'obiettivo di questo lavoro è stato costruire uno strumento per valutare le prestazioni di un particolare layout di magazzino automatico. Un magazzino che avesse massima flessibilità sulle dimensioni, in termine di numero di livelli e numero di canali, e sul numero di diverse tipologie gestibili dal sistema. Scopo principale è stato garantire al futuro utente un'esperienza che possa rispondere ad ogni esigenza di personalizzazione. Inoltre, il modello di simulazione costruito rende possibile sia l'analisi di dati riguardanti direttamente il movimento degli oggetti nel magazzino, come tempo ciclo, throughput e tempo di risposta del sistema in ingresso ed uscita, sia un'analisi energetica sul consumo dei veicoli coinvolti e la loro percentuale di utilizzo sul tempo totale di simulazione, affinché si possano identificare gli elementi più deboli e da perfezionare in un ipotetico dimensionamento di un impianto logistico.
Infine, per agevolare l'utilizzo anche ai meno esperti in simulazione e, in particolare, nell'utilizzo del software scelto, Flexsim, è stata implementata un'interfaccia utente che guida l'utente in ogni passaggio, dalla costruzione del magazzino fino all'avvio della simulazione.

1. Stato dell'arte e simulazioni ad eventi discreti

Elementi centrali del sistema logistico di qualsiasi azienda sono i magazzini, i quali devono essere dimensionati correttamente, in base alle funzioni da assolvere. Dimensionare un magazzino significa identificare quale sia la conformazione migliore affinché il sistema possa fornire una risposta veloce alla richiesta di uno stoccaggio o di una ripresa di un prodotto. Nell'ambito del *material handling*, oltre alla ricerca di un layout migliore, è fondamentale l'implementazione di un sistema di convogliamento che permetta un collegamento tra i magazzini, nel senso vero e proprio di scaffalature, e l'ambiente esterno.
Ad oggi, il sistema più usato per portare unità di carico dalle baie di carico al punto di stoccaggio, o viceversa, è quello in cui sono previsti dei trasloelevatori. Si compongono di un telaio mobile lungo una colonna verticale, con la possibilità di traslare lungo il corridoio tra due scaffalature.

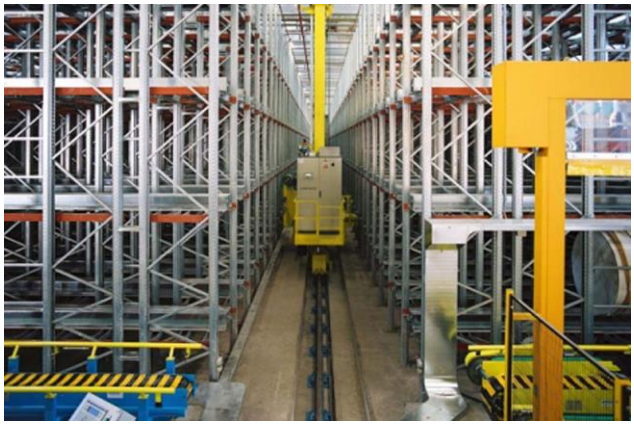

Figura 1 Sistema con trasloelevatori

Gli Automated Storage and Retrieval System (AS/RS) utilizzano questo mezzo di trasporto, muovendosi in modo longitudinale, lungo il corridoio, verticale e trasversale per stoccare o recuperare un'unità di carico, di qualsiasi tipo come pallet o pacchi. Questa struttura presenta sicuramente dei vantaggi rispetto ai sistemi di movimento precedenti, in cui trovavano spazio carrelli elevatori e nastri trasportatori, in particolare:

- Maggiore flessibilità del sistema, capace di raggiungere altezze e lunghezze maggiori in tempi minori e quindi un miglior utilizzo dello spazio a disposizione;
- Sostituzione semplice dei mezzi, con una riduzione dei tempi di fermo macchina;
- Riduzione del costo di lavoro.

Oltre un considerevole investimento inziale, tale sistema presenta, anche degli altri aspetti negativi, poiché è possibile trasportare un solo collo alla volta e la scaffalatura servita può avere una profondità massima di uno o due unità di carico. Per superare questi problemi, si sono sviluppate altre due tecnologie. La prima, CBAS/RS (Crane-Based Automated Storage / Retrieval System) è costituito da nastri trasportatori, scaffali e gru. In particolare, le gru sono completamente automatizzate, con la possibilità di viaggiare tra i corridoi, che dividono le varie scaffalature, per immagazzinare o prelevare i prodotti richiesti dal sistema di gestione. Altro vantaggio, non da poco, è costituito dalla capacità di movimentare unità di carico di diversa dimensione e peso.
Il secondo sistema è l'AVS/RS (Autonomous Vehicle Systems / Retrieval Systems), che sfrutta i progressi ottenuti nella progettazione hardware dell'intelligenza dei veicoli automatici, in modo da rendere facilmente accessibili tutte le posizioni del magazzino e cambiare velocemente in numero dei veicoli coinvolti. La forza di questi sistemi è stata, negli anni, valutata attraverso modelli matematici analitici e simulazioni ad eventi discreti, per dimensionare il magazzino stesso e scegliere le strategie di gestione migliori.

1.1 Autonomous Vehicle Storage and Retrieval System

Nella letteratura scientifica, diverse pubblicazioni si sono concentrate sui sistemi AVS/RS, con lo scopo di valutare le loro performance con modelli analitici, da validare successivamente con software di simulazioni. La prima analisi degna di nota è stata condotta nel 2002 da Malmborg [2], il quale si è soffermato sui parametri migliori per supportare e valutare le scelte di questi sistemi. In particolare, si suggerisce di utilizzare il tempo ciclo delle operazioni, sia di storage che di retrieval, l'utilizzo dei veicoli ed il volume di produzione, considerando le dimensioni del magazzino e le caratteristiche dei veicoli stessi.
Nel 2007 Kuo et al. [3] hanno cercato la spiegazione alla diffusione dei sistemi ULS/RS (Unit Load Storage and Retrieval Systems) nei costi e le difficoltà da affrontare per un'azienda nell'introduzione di tali sistemi, contrapposti al derivante risparmio su tutta la linea.
Nel 2010, Ekren et all. [4, 1] hanno convertito la teoria in esempi pratici. Infatti, hanno studiato un magazzino francese: dopo aver scelto il numero ottimale di ascensori e veicoli, attraverso un software di simulazione hanno identificato i parametri fondamentali per valutare la performance del sistema. Dopo questa prima applicazione, i risultati ottenuti sono stati applicati a diverse realtà industriali, con l'obiettivo di delineare ed analizzare possibili variazioni nei fattori più importanti. Le regole ed i

passaggi di questi esperimenti sono chiamati DOE (Design of Experiment), in ci si specificano le regole di schedulazione delle operazioni ed alcune semplici regole di gestione, come le baie di deposito e ritiro degli item. Gli aspetti valutati, nello specifico, sono il tempo medio di utilizzo degli ascensori e dei veicoli ed il tempo ciclo medio per le operazioni di deposito e prelievo degli item dalle scaffalature.

Successivamente Eken ed Heragu [5], utilizzando un programma di simulazione, hanno confrontato un sistema AVS/RS con un sistema CBAS/RS (Crane-Based Automated Storage and Retrieval Systems), in cui si utilizzano le gru per i movimenti nel magazzino. In particolare, il confronto si è basato su 5 parametri fondamentali:

1. Tempo medio del flusso;
2. Utilizzo dei veicoli coinvolti;
3. Tempo medio di attesa dei veicoli;
4. Numero medio di processi in attesa dei mezzi in coda;
5. Costi di istallazione e gestione.

Le stime delle performance di questi sistemi ha evidenziato la maggior funzionalità, rappresentata dei primi quattro fattori, dei sistemi AVS/RS, i quali risultano più efficienti. Però, tenendo conto anche dei costi, la scelta migliore per un magazzino risultava essere anche la più semplice soluzione con le gru. Spinti da questi risultati, i due autori, in un articolo successivo [6], si sono concentrati sull'ottimizzazione del rendimento del sistema AVS/RS in un magazzino, al variare del numero dei veicoli coinvolti.

A tale risultato, però, sono giunti in modo più veloce Herag ed al. [7] sfruttando l'MPA, Manufacturing Performance Systems Analyser, strumento di analisi in grado di valutare agevolmente diverse setup dei due sistemi, valutandone le prestazioni nei processi di produzione. Negli anni, considerando definitivo l'utilizzo di ascensori e shuttle, si è discusso di quale fosse la gestione più adeguata e convincente per questi veicoli, abbastanza veloci. In particolare, nel 2015, Zou et al [8] hanno introdotto una metodologia, detta *tier-captive,* in cui ogni shuttle può viaggiare e servire solo determinati livelli. Dunque, per ogni operazione di stoccaggio sono da considerare solo due movimenti: uno verticale dell'ascensore ed uno orizzontale dello shuttle dedicato. I risultati ottenuti hanno validato questo metodo per magazzini con canali non troppo profondi e soprattutto un numero massimo di livelli pari a dieci. In magazzini più grandi, non si sono notati considerevoli vantaggi e, anzi, le performance del sistema diminuiscono vistosamente.

Una modifica alla configurazione base dei sistemi AVS/RS è stata sviluppata in una pubblicazione del 2017. D'Antonio et all [9] hanno considerato un terzo veicolo, il quale è scaricato dallo shuttle all'ingresso del canale e si occupa del deposito o il ritiro dell'item. In questi modelli, in cui il movimento è ulteriormente diviso, si possono considerare e valutare:

- Differenti soluzioni di layout, con rack di diverse dimensioni;
- Diversi criteri per lo storage ed il retrieval;
- La possibilità per shuttle e satellite di effettuare diverse operazioni contemporaneamente e, dunque, la parallelizzazione delle azioni;
- Calcolare media e deviazione standard del tempo ciclo.

A rafforzare questi concetti, sempre nel 2017, Lerher [10] ha identificato i fattori caratterizzanti le performance di un sistema, soprattutto per ottimizzare il tempo medi di utilizzo dei veicoli, i throughput e la capienza del magazzino. Nello specifico sono importanti dati strutturali del magazzino, come il numero dei livelli ed il numero dei canali per ogni livello, e le caratteristiche tecniche dei veicoli, cioè accelerazione e velocità dell'ascensore, degli shuttle e dei satelliti.

1.2 Simulazioni ed eventi discreti

Le simulazioni sono modelli consolidati che aiutano a comprendere la complessità di un sistema reale [11]. Inoltre la simulazione è un valutatore di soluzioni, non un generatore di soluzioni: non produce una teorica ottima soluzione ma piuttosto cerca la miglior possibile soluzione [12].
Un sistema è una serie di entità, come persone o macchine, che agiscono ed interagiscono insieme per il compimento di un determinato fine logico [13].
Importante è definire lo *stato* di un sistema [14], cioè la collezione di variabili necessarie per descrivere un sistema in un particolare istante di tempo. In particolare, i sistemi possono essere:

- *Discreti,* in cui lo stato delle variabili cambia istantaneamente in un determinato punto del processo;
- *Continui,* in cui lo stato delle variabili cambia continuamente nel tempo.

Il termine *simulazione* indica la riproduzione di un determinato comportamento attraverso logiche ed algoritmi matematici. Sulla scena della produzione industriale, da qualche anno, giocano un ruolo

importanti le soluzioni offerte da alcuni software di simulazioni ad eventi discreti. Questo tipo di simulazioni sono usate per modellare sistemi, il cui stato cambia in determinati istanti temporali a seguito di un determinato evento. Il loro punto di forza è nel poter ricostruire in modo fedele tutte le soluzioni industriali presenti sul mercato, per poterne analizzare vantaggi, svantaggi e possibili miglioramenti. Inoltre, con una simulazione, si abbatte il vincolo temporale, potendo verificare il funzionamento di un sistema simulando il suo comportamento nel tempo. Dunque, una simulazione può essere uno strumento predittivo, quindi utilizzabile per validare un impianto produttivo prima della vendita o dell'istallazione, o uno strumento di correzione di alcune dinamiche negative nel quotidiano funzionamento di un sistema.
Solitamente un modello di simulazione è una determinata e, spesso, radicale astrazione di un sistema reale che può essere utilizzato per rispondere a domande o risolvere problemi [15].
In generale, in un modello sono presenti molti elementi, tra cui i più importanti sono:

- *Variabili globali,* descrivono alcune informazioni o caratteristiche del modello, sono utili negli algoritmi decisionali sia in lettura che in scrittura;
- *Eventi*, circostante che condizionano il modello e possono cambiare il valore di una variabile;
- *Entità,* elementi del sistema di cui definire il comportamento;
- *Attributi,* informazioni da assegnare alle varie entità nel modello, per identificarle e poterle raggruppare per tipologia;
- *Risorse,* elementi con cui alimentare un servizio alle entità;
- *Attività,* azioni a cui si attribuisce una durata prima della sua realizzazione, che può essere costante o soggetta alle regole delle distribuzioni matematiche;
- *Ritardi,* archi temporali utilizzati per rallentare alcune operazioni o per descrivere tempi di fermo del sistema.

Quando si approccia con delle simulazioni ad eventi discreti, ci si pone sempre la domanda di quale sia il livello di dettaglio giusto per il proprio modello. È chiaro come inserire dettagli minuziosi può giocare all'aspetto grafico della simulazione, immagazzinando però un'elevatissima quantità di dati che rallenta la reattività del calcolo computazionale del software utilizzato. Quindi è consigliato, e molte volte anche necessario, utilizzare nel modello solo gli elementi indispensabili per simularne il funzionamento ed il comportamento nel tempo. In particolare, in un suo lavoro, Barnett ha attribuito quattro dimensioni fondamentali per un modello [16] :

1. *Sistema di interesse,* cioè la tipologia di informazione in output dalla simulazione. Nella produzione si potrebbe essere interessati alla movimentazione del materiale, alla gestione di un magazzino, al tempo ciclo di una linea. Altro comportamento interessante è la variabilità di questi parametri e la loro flessibilità nell'adattarsi rapidamente ed adeguatamente a situazioni particolari del sistema;
2. *Visibilità,* definendo un modello come:
    - *trasparente*, cioè che rappresenta nello specifico tutti i processi nel modello usando leggi fisiche documentate;
    - *black-box*, in cui non si rappresentano chiaramente le dinamiche interne ma ci si limita, con una certa efficienza, a trasformare i dati in ingresso in un risultato finale in uscita;
3. *Probabilità,* cioè:
    - *Modello probabilistico*, i dati in input si riflettono in diversi output finali con variazioni descrivibili statisticamente;
    - *Modello Deterministico,* i risultati in output non presentano influenze statistiche o variazioni e sono determinati univocamente a partire dal set di input in ingresso, poiché il modello non contiene nessun elemento probabilistico.

2. Costruzione del modello

Il modello ha la struttura di un magazzino in cui sono presenti uno o due rack e il sistema di movimentazione AVS/RS, composto da un ascensore, gli shuttle e i satelliti. Nello specifico è usata una configurazione *tier-to-tier,* cioè i veicoli sono liberi di lavorare su ogni livello, con l'unico vincolo di non contemporaneità sullo stesso livello. In contrapposizione, esiste la logica *tier-captive,* in cui ad ogni livello è assegnato un unico e determinato shuttle. In figura è schematizzato l'idea di partenza del modello nella sua interezza: nella parte frontale, all'ingresso delle scaffalature si trova l'ascensore (*lift*), nei corridoi lungo i livelli agiscono gli shuttle e nei canali i satelliti.

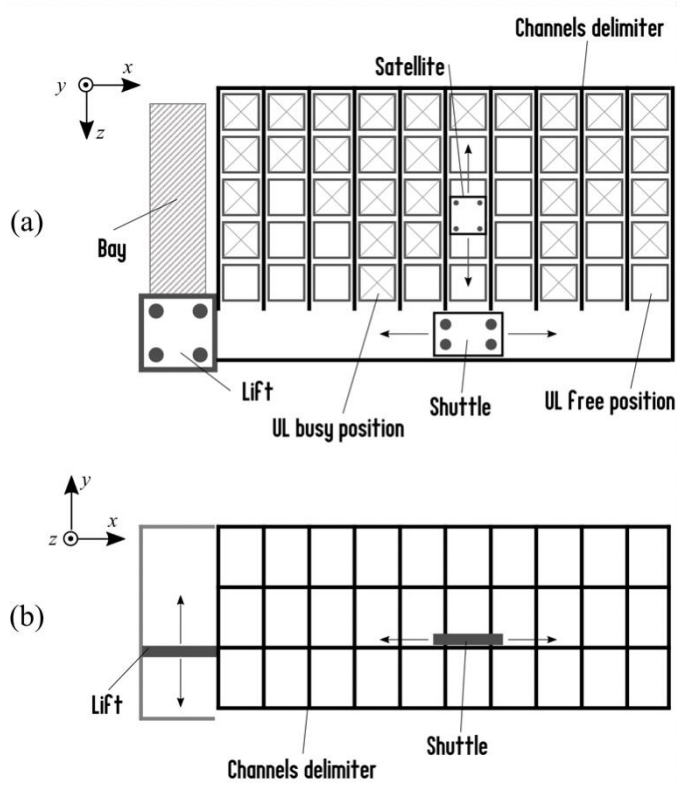

Figura 2 Schemi sistema AVS/RS

Un' impianto del genere è strutturato per asservire operazioni di *storage*, cioè deposito in magazzino degli oggetti, denominati successivamente *item*, ed operazioni di *retrieval,* ripresa di tali oggetti per un'apposita richiesta del sistema. Prima di approcciarsi alla simulazione, è necessario focalizzare l'attenzione sui problemi riscontrabili, tenendo conto dei vincoli del sistema e tutte le possibili combinazioni di eventi. Un aiuto in questa attività è realizzare un diagramma UML, in cui analizzare ogni step, sia del processo di storage che di retrieval, ed identificare le situazioni che potrebbero verificarsi. I diagrammi sono divisi in quattro sezioni, la prima dedicata al sistema decisionale e le altre dedicate alle tre classi di veicoli. Per lo storage, le operazioni iniziano con l'arrivo della richiesta

e la decisione della destinazione. Ogni blocco rappresenta un'attività ed il flusso appare lineare fino al completamento del deposito in magazzino.

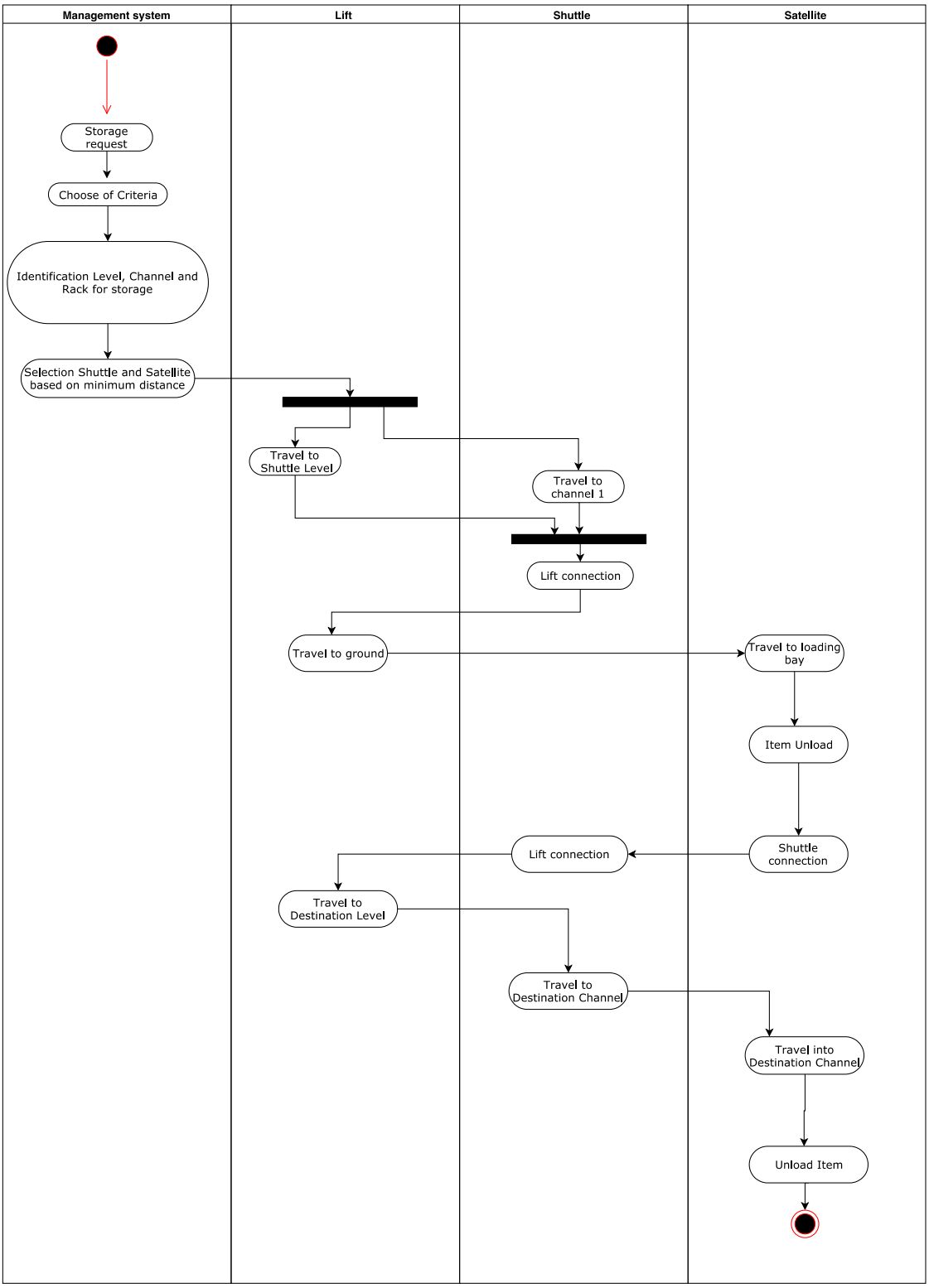

Figura 3 Diagramma UML Storage

La struttura per il retrieval è analoga. Dopo la decisione dell'allocazione dell'item da ritirare, possono verificarsi diverse situazioni, da analizzare separatamente. Infatti, è visibile come sia presente un blocco decisionale e diverse operazioni parallelizzate.

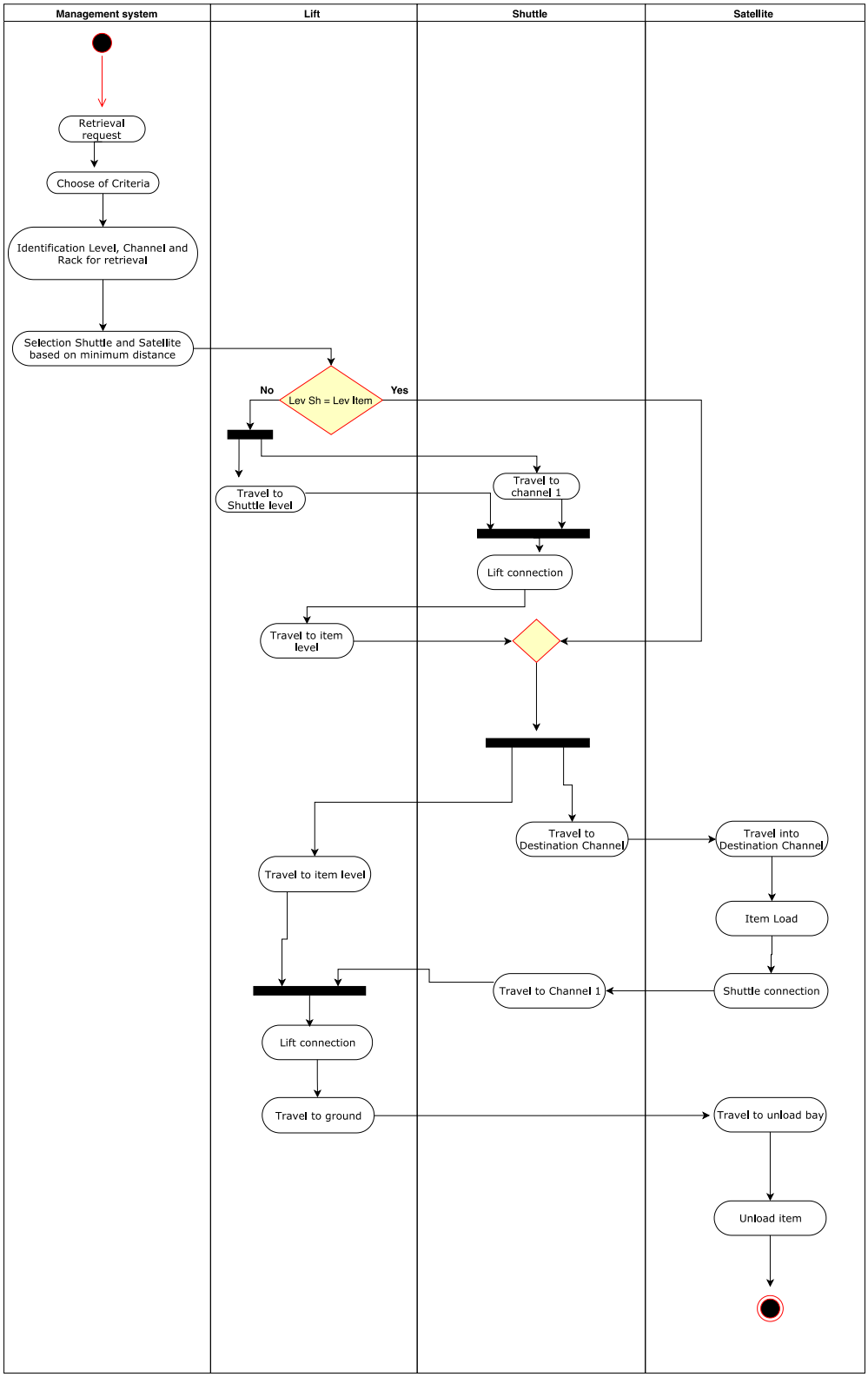

Figura 4 Diagramma UML Retrieval

Questi diagrammi sono dei punti di partenza per affrontare la programmazione della simulazione, il cui compito fondamentale è esplodere i blocchi di attività e rendere contemporanee i due processi, sia come management delle decisioni sia come movimenti veri e propri dei veicoli.

2.1 FlexSim

FlexSim è un software di simulazione sviluppato dalla FlexSim Software Products, Inc., e distribuito in Italia esclusivamente dalla Flexcon S.R.L. È un potente strumento di analisi che aiuta gli ingegneri ed i progettisti nel prendere decisioni intelligenti nell'ideazione e nel funzionamento di un sistema di produzione. [17]

Figura 5 Logo Flexsim

Flexsim permette la creazione 3 D di tutte le soluzioni industriali possibili. Oggi, infatti, visto l'elevato costo delle attrezzature necessarie per la realizzazione di linee produttive o magazzini logistici, assume un ruolo sempre più importante la simulazione a priori di tali sistemi, con lo scopo di analizzare e scegliere la soluzione migliore. FlexSim è la risposta a questi problemi, poiché con la sua grafica tridimensionale è possibile creare un modello virtuale, molto fedele alla realtà. La creazione di oggetti può avvenire in due modi. Il primo, più semplice ed intuitivo, sfrutta la vasta scelta proposta dalla libreria implementata nel software: vari elementi, come operatori, sorgenti, macchine o baie, possono essere trascinate nel pannello grafico. Si può sia connettere tra di loro i vari elementi, sia impostare diverse impostazioni e comportamenti in diverse situazioni, come ad esempio un carico/scarico di un item o la fine di un processo di una macchina o semplicemente l'ingresso/uscita di un item da una baia di carico.

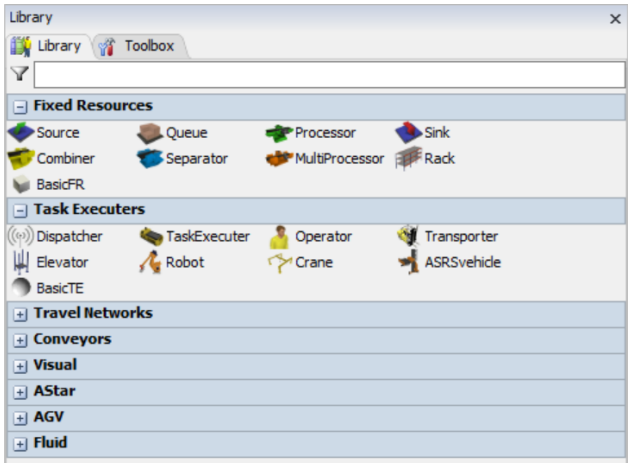

Figura 6 Libreria oggetti Flexsim

Un ulteriore punto di forza dell'ambiente Flexsim è la grande personalizzazione del modello, sia come oggetti che come logiche implementabili. Infatti, sia gli oggetti, che le varie decisioni, sono implementabili attraverso script di codice, nel linguaggio denominato *Flexscript*, in cui le regole base richiamano il linguaggio di programmazione C++. Infatti, il modello di studio è stato creato attraverso la console del software.

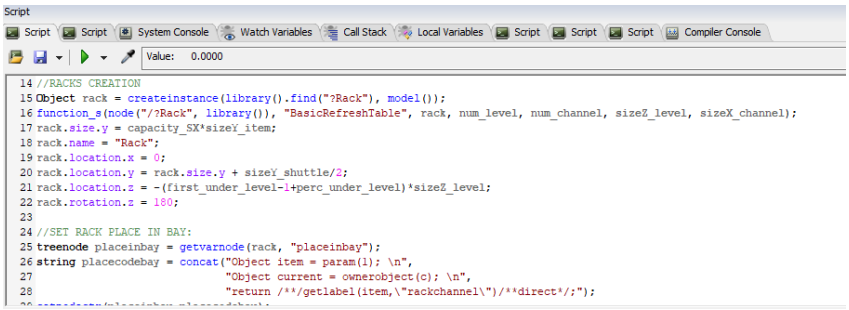

Figura 7 Script Console Flexsim

2.2 Topologia magazzino

La realizzazione del magazzino è stata effettuata attraverso uno script, in cui sono stati creati gli oggetti necessari, assegnandogli una collocazione nello spazio attraverso coordinate spaziali (asse x, asse y ed asse z), ed i collegamenti tra loro. Lo script si è reso necessario per non contraddire lo scopo principale di questo lavoro di tesi, cioè rendere la struttura più flessibile possibile per un futuro utente, il quale potrà modificare alcune caratteristiche del magazzino in modo veloce, cambiando il solo valore numerico di alcune *variabili globali.*

2.2.1 Rack

La prima decisione da prendere riguarda il numero di rack. Il modello prevede due soluzioni possibili: la presenza di una sola scaffalatura o di due scaffalature simmetriche con un corridoio centrale in cui avverrà il movimento dei veicoli trasportatori degli item. Ogni rack si svilupperà in altezza, identificando così vari livelli, ed in profondità, identificando un certo numero di canali per ogni livello. Il vincolo di simmetricità impone lo stesso numero di canali per ogni livello. Nel caso di due rack, essi avranno le caratteristiche, ad eccezione della capacità massima di ogni canale. Dunque, attraverso le variabili globali, è possibile variare:

- Il numero dei canali;
- Il numero dei livelli;
- La capacità massima di ogni canale, differenziandola eventualmente tra i due rack;
- Le dimensioni strutturali, dipendenti dall'altezza, lunghezza e profondità di ogni canale.
- La presenza di livelli interrati. C'è la possibilità di indicare il primo livello posto sotto il pavimento e la percentuale di interramento.

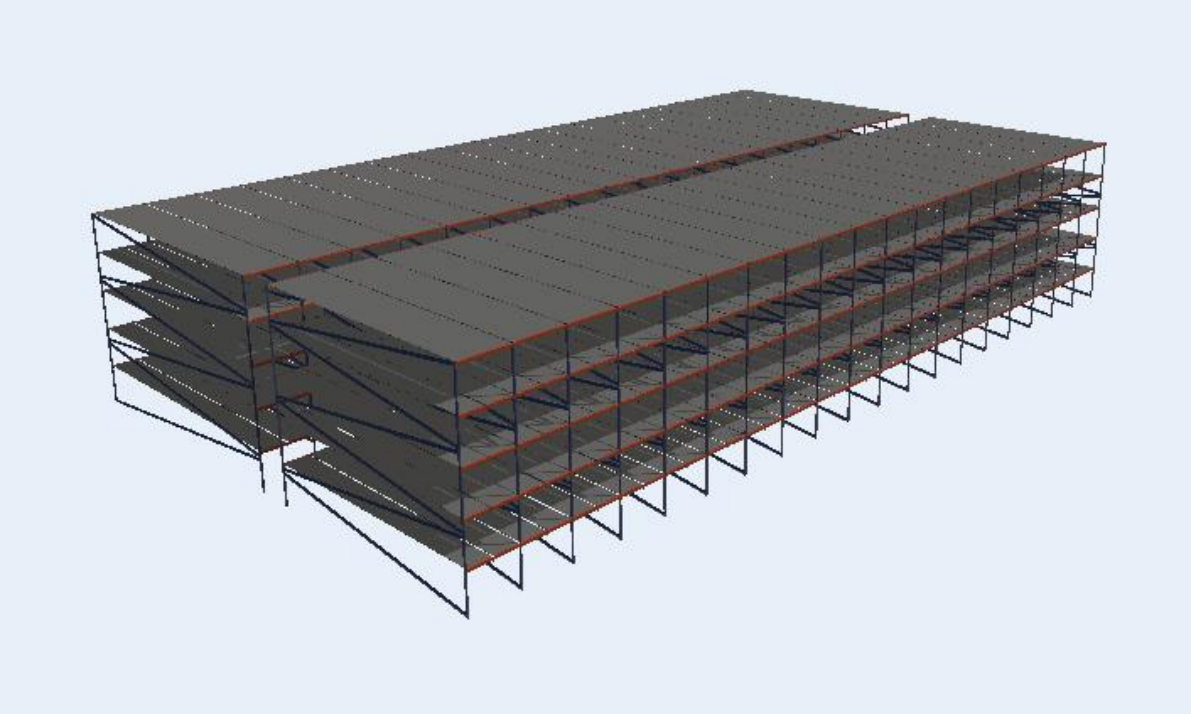

Figura 8 Rack

2.2.2 Veicoli

I veicoli previsti dal modello sono: ascensore, shuttle e satellite. L'ascensore è unico e posto centralmente, in corrispondenza del corridoio centrale tra i due rack. Il suo compito è di trasportare l'accoppiata shuttle/satellite tra il pavimento, in cui avviene il carico/scarico degli item, ed i livelli, spaziando dunque l'asse z del modello tridimensionale. Gli shuttle, invece, trasportano il satellite lungo un determinato livello, muovendosi lungo l'asse x e fermandosi in corrispondenza del canale in cui si è stabilito di effettuare un'operazione di storage o retrieval. Tale operazione è svolta, infine, dal satellite, il quale si muove all'interno del canale (asse y), raggiunge la posizione in cui depositare o ritirare l'item e torna indietro per riaccoppiarsi con lo shuttle. Per ragioni di sicurezza, in ogni livello può agire contemporaneamente una sola coppia di shuttle/satellite, dunque il numero di questi veicoli è variabile, deciso dall'utente ma non può essere superiore al numero dei livelli.
Le traiettorie dei veicoli sono governate da una serie di nodi, creati anche essi nello script inziale. In particolare, si identificano per primi i nodi destinati al movimento dell'ascensore, posizionati alla quota esatta del livello della scaffalatura in modo che si fermi in posizione corretta. Da questi nodi, posizionati lungo l'asse z, scaturiscono gli altri nodi lungo l'asse x, i quali determinano l'ingresso dei vari canali per ogni livello e sono i responsabili delle traiettorie degli shuttle. Per ogni veicolo è possibile modificare:

- La velocità e l'accelerazione;
- Il tempo di accoppiamento tra i diversi veicoli;
- Il tempo di carico/scarico degli item.

Figura 9 Veicoli nel modello

2.2.3 Sorgenti e baie

Nel modello sono previste 5 sorgenti:

- *SourceStorage,* in cui sono creati gli item che devono subire un processo di storage;
- *InitialWallDx e InitialWallSx,* in cui sono creati degli item, identificati come tipologia "0" e di color nero, rappresentanti degli impedimenti nella struttura del magazzino (come ad esempio una colonna portante). Dunque, in questo modo, si può limitare la capacità di un determinato canale. Logicamente questi item non potranno essere mai presi dalle loro posizioni, cioè non subiranno mai retrieval.
- *InitialDx e InitialSx,* in cui sono creati gli item rappresentanti la situazione iniziale, cioè gli item già presenti al tempo 0 di inizio simulazione.

Le logiche di funzionamento di ogni sorgente saranno spiegate in dettaglio nel prossimo paragrafo. Inoltre, nel modello sono presenti due baie:

- *StorageBay,* in cui si accumulano gli item in attesa di essere stoccati;
- *RetrievalBay,* in cui vengono depositati gli item che hanno subito un retrieval.

Nella figura seguente, è possibile vedere come appaiono gli oggetti dopo l'esecuzione dello script.

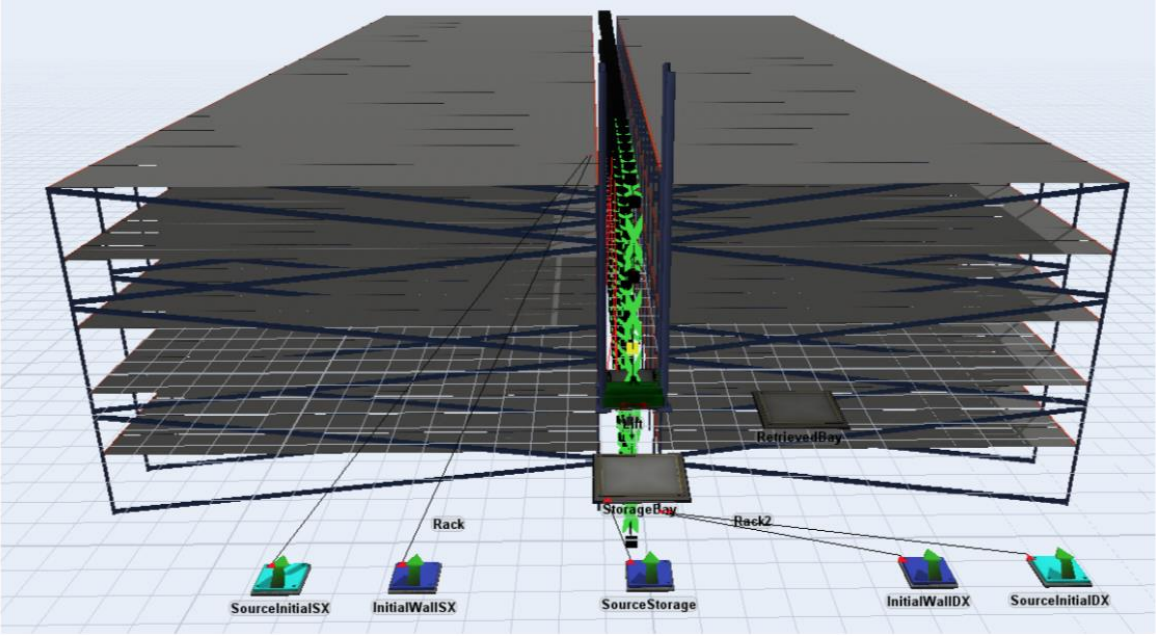

Figura 10 Oggetti del modello

2.3 Process Flow

Nel software Flexsim è presente un modulo molto utile, denominato Process Flow. Come indica il nome, in questa sezione è possibile ricostruire graficamente l'intera logica del sistema di simulazione, attraverso una libreria che comprende diversi blocchi di attività.

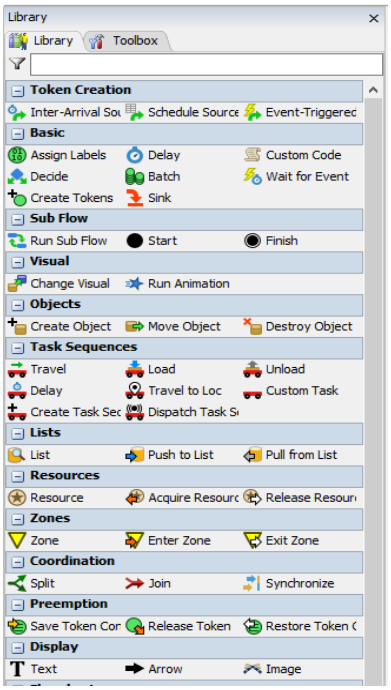

Figura 11 Libreria Process Flow di Flexsim

Il Process Flow rappresenta il vero cervello del programma. Infatti, è possibile selezionare le attività ed organizzarle nello spazio, riunendole anche in gruppo. I vari collegamenti determinano il percorso da seguire per ogni *token.* Un *token* è l'elemento rappresentante lo stato della simulazione, eseguendo nell'ordine le varie attività. È rappresentato visivamente da un cerchietto verde, che man mano prosegue nelle varie fasi. In particolare, nel modello sono stati utilizzati:

- ***Token Creation***, attività di creazione dei token.

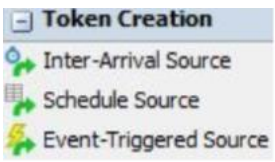

Figura 12 Libreria Process Flow: Token Creation

1. *Inter-Arrival Source*, la creazione dei token segue una distribuzione del tempo, che può essere uniforme o esponenziale.
2. *Schedule Source,* la creazione dei token segue una schedulazione, che può essere inserita manualmente o caricando un file Excel. È possibile indicare il tempo della creazione ed alcune caratteristiche da salvare in ogni token.
3. *Event-Triggered Source*, il token è creato ed inizia a seguire il percorso del process flow solo in seguito ad un determinato evento durante la simulazione.

- ***Basic***, attività di base per decidere il comportamento del token.

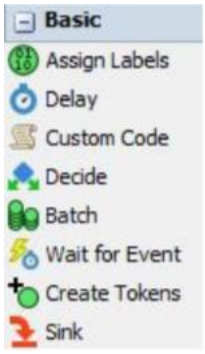

Figura 13 Libreria Process Flow: Basic

1. *Assign Label,* in cui è possibile assegnare ad ogni token delle informazioni di valore numerico, attraverso script. Ogni informazione resterà memorizzata nel token e sarà possibile interrogarla ogni volta sia necessario.
2. *Delay,* indica un tempo di attesa. Può essere un intervallo di tempo determinato, una distribuzione uniforme o esponenziale.
3. *Custom Code,* rappresenta un codice script, in cui poter codificare qualsiasi azione. Ad esempio, si potrebbe modificare una variabile globale o il valore della cella di una tabella. La potenza degli script è illimitata, governata da una scrittura del tipo "C++" e dalle sue funzioni base ("ciclo for", "condizione if").
4. *Decide,* bivio di decisione per ogni token. Solitamente è implementato attraverso uno script e decide la direzione da assegnare al token, in base al verificarsi di opportune circostanze.
5. *Wait for Event,* induce il token a fermarsi ed aspettare l'accadimento di un certo evento;
6. *Sink*: elimina il token e tutte le informazioni in esso contenute.

- ***Task Sequences****,* attività assegnate ai task executer, indicanti precise azioni da compiere.

Figura 14 Libreria Process Flow: Task Executer

1. *Travel*, fa viaggiare il task executer verso una precisa destinazione, indicata da un oggetto.
2. *Load*, indica il caricamento di un oggetto della simulazione sopra un task executer.
3. *Unload*, indica lo scarico di un oggetto della simulazione da un task executer.
4. *Delay*, indica un ritardo dell'attività successiva.
5. *Travel to Loc*, simile al Travel, ma con indicazione spaziale della destinazione.

- ***Lists****,* attività per creare e gestire liste del modello, in cui inserire token, task executer o item. Sono presenti solo due funzioni, *Push to List* o *Pull from List,* per aggiungere o prelevare dalla lista oggetti.

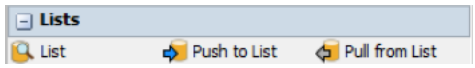

Figura 15 Libreria Process Flow: Lists

- **Resources**, per acquisire o rilasciare una risorsa creata nel modello, come ad esempio l'ascensore.

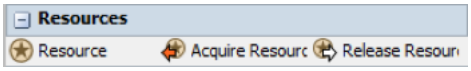

Figura 16 Libreria Process Flow: Resources

- **Coordination**, molto utile per parallelizzare le attività del process flow.
    1. *Split*, divide il token in due copie perfettamente uguali che eseguiranno attività diverse contemporaneamente.
    2. *Join*, riunisce i due token nati dallo *split.* Solo all'arrivo dei due token gemelli, il token inziale sarà rilasciato e le attività potranno riprendere.

Figura 17 Libreria Process Flow: Coordination

Il Process Flow definitivo del modello è mostrato nella figura successiva. È possibile dividerlo in blocchi, in cui ci sono le decisioni determinati per le logiche decisionali implementate.

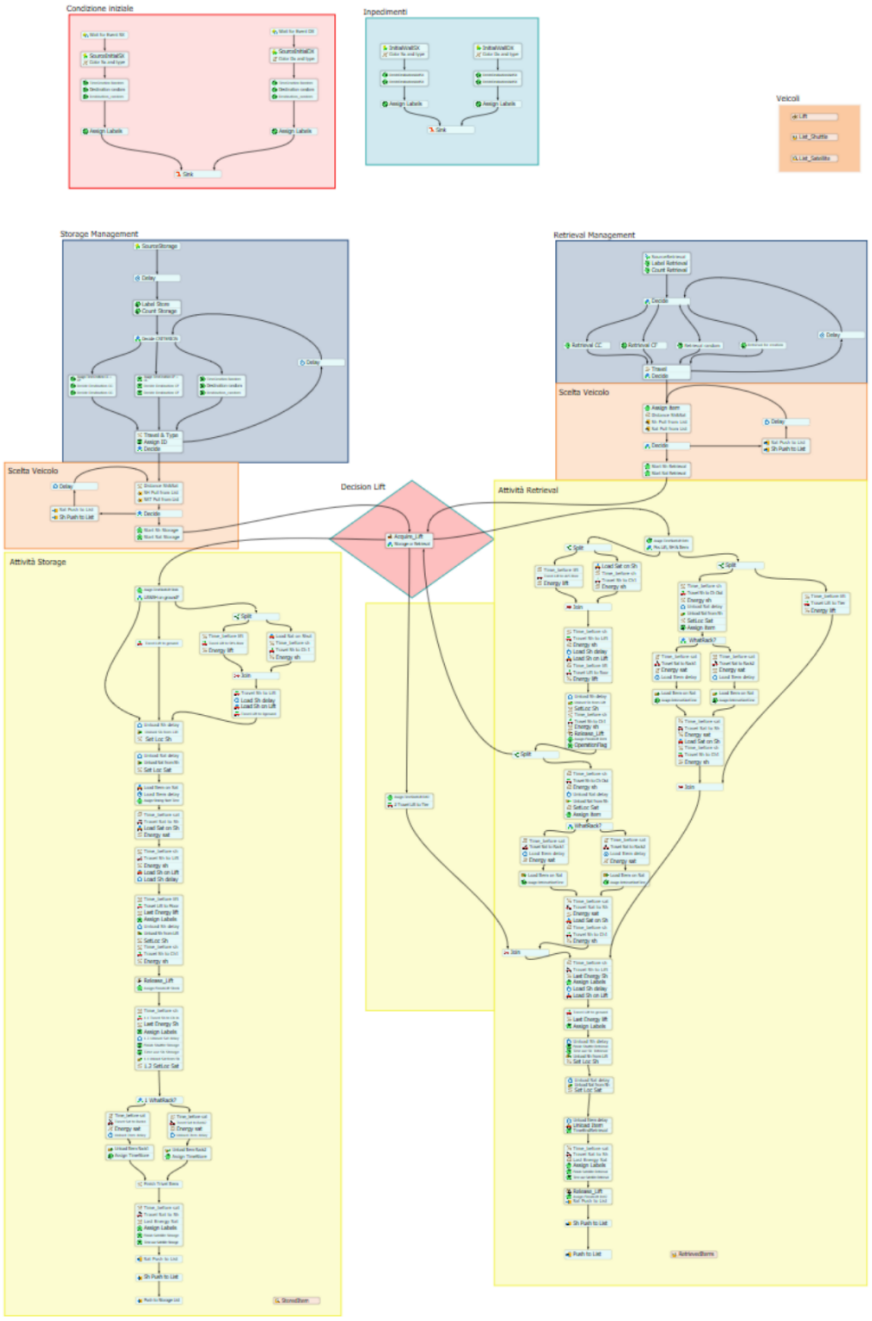

Figura 18 Process Flow del modello a blocchi

2.4 Rappresentazione impedimenti

Nell'ordine di avere un magazzino più personalizzabile possibile, è possibile creare degli impedimenti interni alla struttura delle scaffalature stesse, rappresentanti una riduzione di capacità di determinati canali. Nella realtà industriale, questi impedimenti potrebbero rappresentare delle colonne portanti nella struttura o delle posizioni inaccessibili. Inoltre, in questo modo, è possibile creare rack con strutture asimmetriche, con canali aventi diverse capacità massime. Nella pratica del software, è necessario importare una tabella per ogni rack (destro e sinistro). Ogni elemento delle matrici, costruite su Excel, indica univocamente un canale, in quanto le righe rappresentano i canali e le colonne sono i livelli. In ogni casella si indica la capacità del canale.
Il metodo più semplice individuato per simulare questa situazione, è stato depositare degli item di colore nero al tempo zero, cioè prima del reale inizio della simulazione, per limitare le capacità dove necessario. Tali item, indicati dal "*Type 0"* non posso subire un processo di retrieval.

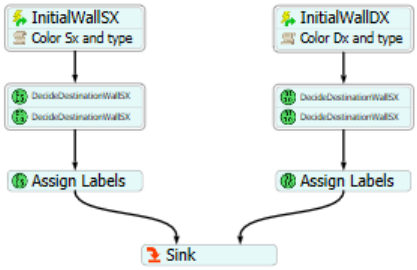

Figura 19 Process Flow: Impedimenti

Nel Process Flow si individuano due sorgenti, una per ogni rack, con un solo arrivo, al tempo zero, di una quantità di item pari alle posizioni inaccessibili rispettivamente per i due rack. Lo script, contenuto nell' *assign label,* denominato "*Decide Destination",* effettua un ciclo per ogni combinazione di livello e canale, verifica se il corrispettivo valore in matrice sia inferiore alla capacità massima del canale e nel caso, invia istantaneamente un item nero in tale canale. Il ciclo termina quando le due matrici sono saturate, cioè quando in ogni casella è presente il valore di capacità massima di un canale per quel Rack. È importante sottolineare che tali azioni sono immediate, senza dispendio di tempo, in modo da non alterare la simulazione. Questo è reso possibile proprio dalle funzioni del Process Flow, in grado di gestire anche sistemi astratti, senza visualizzare graficamente il movimento ma rappresentando solo il risultato voluto.

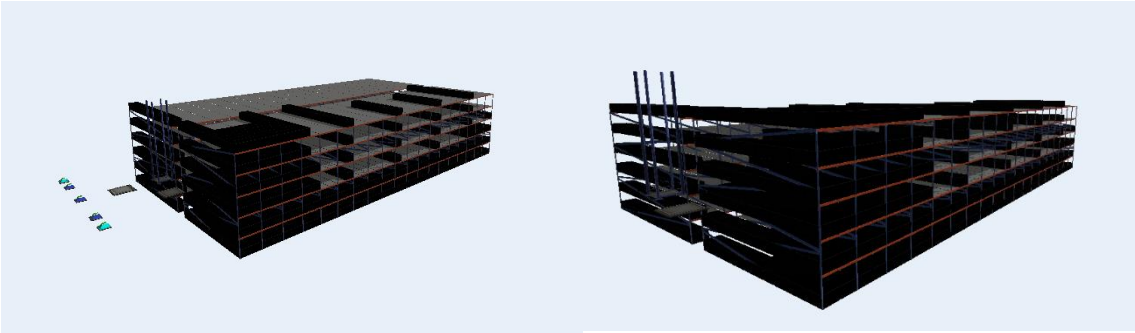

Figura 20 Modello con impedimenti

2.5 Condizione iniziale

Analogamente alla costruzione precedente, anche per le condizioni iniziali troviamo una configurazione a specchio per i due rack. L'ordine di esecuzione è molto semplice, un *Waiting for Event* aspetta il completo inserimento degli item neri, cioè degli impedimenti strutturali. Questa attesa ha un tempo fittizio, in quanto l'operazione precedente termina computazionalmente al tempo 0. Anche qui la creazione dei token è regolata da due sorgenti, ma con un *inter arrival*, di valore atteso molto basso (0.001 secondi) e varianza nulla, non disturbando quindi il tempo generale di simulazione. Il numero di item iniziato deve essere scelto dall'utente, indicato come percentuale della capacità massima del magazzino.

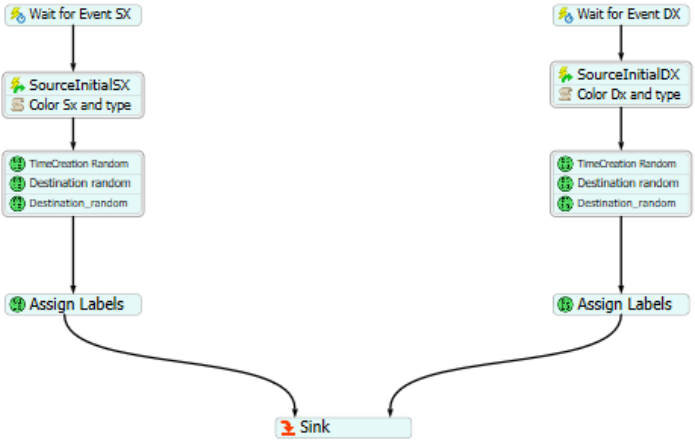

Figura 21 Process Flow: Condizione iniziale

Invece il *custome code,* indicato con "*Color and type",* tiene conto della scelta del numero di tipologie diverse di item, che nella pratica non ha limiti. Ad ogni tipologia sarà assegnato un colore diverso, in modo da rendere visibile la differenza nella visualizzazione 3D. Il criterio di storage utilizzato per la condizione iniziale è randomico, in cui ci sono solo due vincoli:

1. In ogni canale è possibile stoccare una sola tipologia di item;
2. Si riempie ogni canale fino alla capacità massima, prima di identificare un nuovo canale in cui effettuare gli storage.

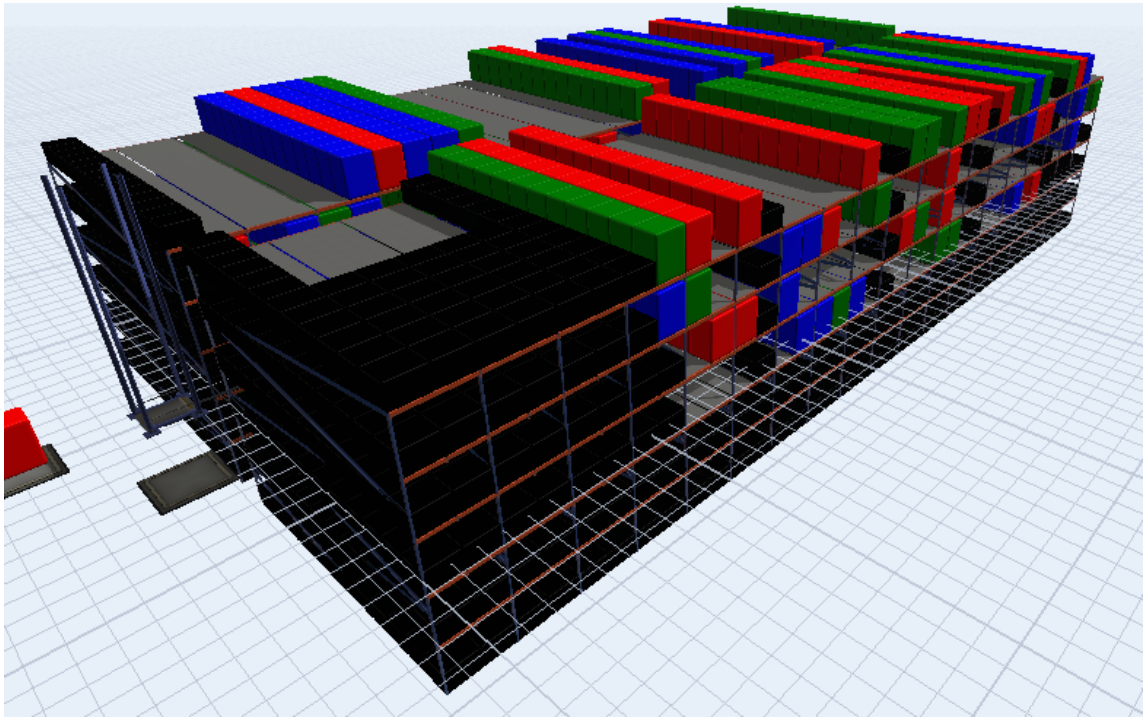

Figura 22 Modello dopo condizione iniziale

In questa vista è possibile notare sia come la scelta dei canali sia completamente casuale, sia che ogni canale contenga una sola tipologia di item, identificabile dal colore.

2.6 Storage Management

Come già detto, l'elemento fondamentale del Process Flow è il token, nel quale è possibile immagazzinare ogni tipo di informazione, durante lo svolgersi delle varie attività. Dunque, è chiaro che il primo elemento del flusso è una sorgente, elemento addetto alla creazione dei token. Nella *Source Storage* è stata implementata una logica d'arrivo "*inter arrival"*, con media e varianza scelta

dall'utente finale. Per quanto riguarda le diverse tipologie di item, si è scelto una distribuzione casuale dei tipi, per verificare il comportamento e le prestazioni del simulatore nelle situazioni più randomiche possibile. Con pochi comandi, però, è possibile cambiare tale impostazione in un arrivo schedulato o sequenziali dei vari item. Le operazioni successive sono un *delay,* molto piccolo, per assicurarci che la simulazione avvenga dopo le operazioni iniziale descritte precedentemente e dei contatori dell'effettivo numero di richieste di storage, ricevute dal sistema.

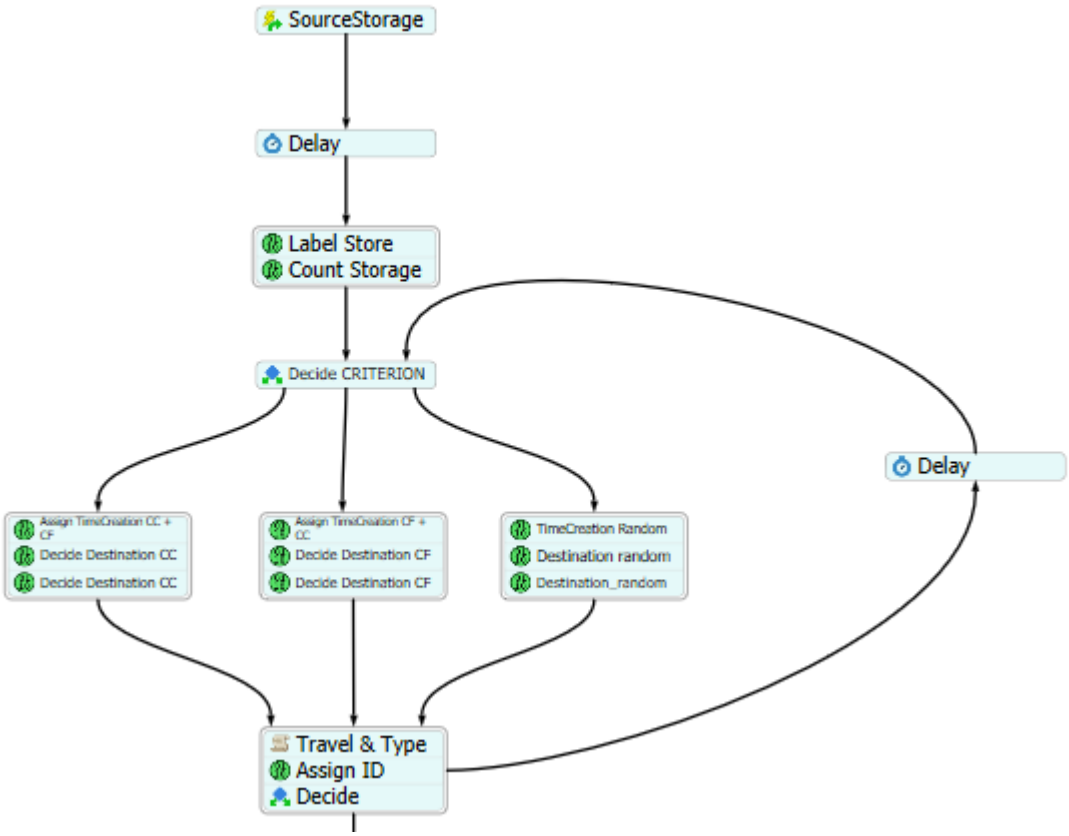

Figura 23 Process Flow: Storage Management

La prima decisione importante riguarda il criterio di storage. I metodi di riempimento di un magazzino possono essere molti e basati su logiche decisionali diverse. In generale le scelte si basano sulle necessità della realtà in cui il magazzino dovrà lavorare e può tener conto di vincoli, come minor spazio percorso, minor tempo di percorrenza o risparmio energetico. Nel modello sono stati implementati tre criteri base, ognuno identificato da una cifra, dai quali è possibile, con pochi cambi nel codice decisionale, sviluppare altri metodi, personalizzabili in base alle diverse esigenze industriali. Per accedere al criterio, il *Decide,* legge la scelta effettuata dall'utente per scegliere il ramo da far seguire al *token* ed in particolare, in base alla cifra, si sceglie:

1. *Criterio Closest Channel;*

2. *Criterio Closest Floor;*
3. *Criterio Random.*

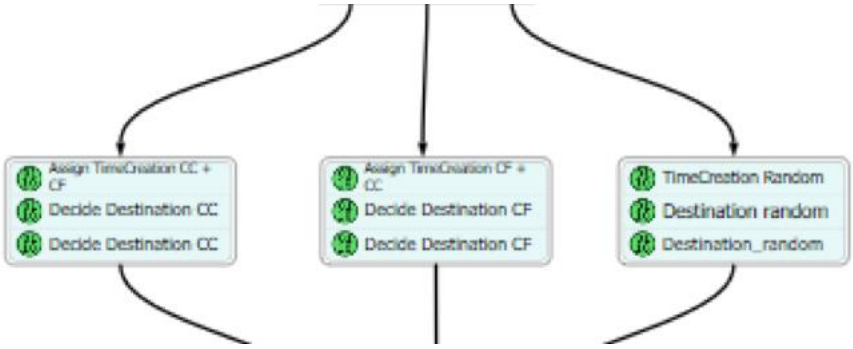

Figura 24 Process Flow: Criteri di storage

Ogni criterio ha un suo blocco decisionale. La struttura è identica per i 3 metodi, con in successione:

- *Assign TimeCreation*, con cui memorizzare nel token l'informazione del tempo esatto di creazione della richiesta;
- 2 *Decide Destination,* il primo memorizza le informazioni di destinazione nel token, il secondo copia tali informazioni anche nella memoria propria dell'item. In questo modo, selezionando uno specifico item, immagazzinato o in movimento, si può risalire velocemente alla sua locazione.

È facile intuire come nella destinazione sono rinchiuse le logiche di gestione del magazzino stesso. Il problema più grande riscontrato nella modellazione è stato riuscire a far coincidere il tempo decisionale con il tempo di simulazione vero e proprio. Nello specifico, *Decide Destination* è la prima attività gestionale di un token dalla sua creazione. In tempi computazionali, ad un token, appena creato, viene subito assegnato il livello, il canale e il rack in cui sarà allocato dopo il processo di storage. Tale assegnazione, seconda diverse logiche per i 3 criteri, sarà identificata da un *array,* composto appunto da tre valori. La scelta del canale prescelto si fonda su due pilastri da non contraddire, vale a dire che in quel canale sia vuoto o ci sia già presente un item dello stesso tipo e che la sua capacità non sia satura, cioè il numero di item già immagazzinato sia inferiore alla capacità massima. Su queste due considerazioni sono sorti i problemi maggiori. Infatti, il software *Flexsim*, ha dei comandi implementati che restituiscono il numero di item in un canale ed il tipo dell'ultimo item, ma effettua dei controlli *real-time*, fotografia della situazione nell'istante in cui è lanciato il comando. Il modello, invece, dopo l'assegnazione della destinazione, simula fedelmente tutte le operazioni da svolgere per lo storage, tra cui l'acquisizione dell'ascensore, il carico dell'item sul

satellite, l'accoppiamento tra i vari veicoli, il viaggio dei veicoli fino al punto di arrivo dell'item. Tutte queste operazioni, visibili chiaramente nella grafica 3 D, hanno dei tempi di esecuzione, con la conseguenza che tra la decisione definita della destinazione di allocazione ed il reale deposito dell'item, intercorrono molti secondi. Inoltre, dopo la prima decisione, gli item trascorrono un tempo di attesa nella baia dinanzi all'unico ascensore, il quale da solo deve asservire sia le missioni di storage che di retrieval, diventando collo di bottiglia di tutto il modello. Nella pratica accadeva abitualmente che, nonostante il controllo attraverso i comandi Flexsim restituisse esito positivo, l'item al momento del deposito si ritrovasse allocato al di fuori del canale, nel corridoio, essendo il canale era già pieno. Dunque, questo problema evidenzia la necessità di implementare delle logiche capaci, oltre a leggere la situazione di ogni canale in tempo reale, di prevedere ciò è già stato deciso e necessariamente accadrà. Per questo motivo sono state inserite delle tabelle genarli, due per ogni rack, create direttamente nello script iniziale, con numero di righe pari al numero di canali e numero di colonne pari al numero di livelli. Come al solito, ogni casella della matrice identifica unicamente un canale.

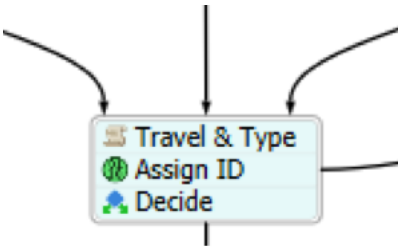

Figura 25 Process Flow: particolare aggiornamento tabelle

Tutte le tabelle sono inizializzate con valori uguali a 0, per ogni casella. Nel blocco successivo al *Decide Destination*, è presente un codice script, denominato *Travel&Type*, in cui si aggiornano proprio tali tabelle: per ogni rack, la prima tabella tiene conto degli item già destinati in un determinato canale, incrementando la casella corrispondente di un'unità; la seconda matrice aggiorna il valore a seconda della tipologia di item immagazzinato. Ora è semplice semplificare i controlli generali effettuati nello script *Decide Destination,* per ogni canale:

- la somma del numero di item presenti momentaneamente nel canale, identificati tramite un comando Flexsim, e il valore letto nella prima matrice, deve essere inferiore alla capacità massima del canale stesso;

- la tipologia dell'item da stoccare deve essere la stessa della tipologia già presente nel canale, informazione letta nella seconda matrice. Al massimo è possibile stoccare in un canale in cui è identificato il "tipo 0", sinonimo di canale vuoto o con impedimenti strutturali.
-

In questo blocco di attività, si assegna anche un ID all'item, in modo da renderlo sempre identificabile. Molto importante è il *decide* finale, poiché copre l'ultima situazione possibile, alla richiesta di uno storage, l'assenza di posti disponibile, cioè un magazzino completamente pieno. In questo occorrenza, lo script precedente, *Decide Destination,* assegna all'array tutti valori negativi, impossibili ed il bivio decisionale invia il token in un loop.

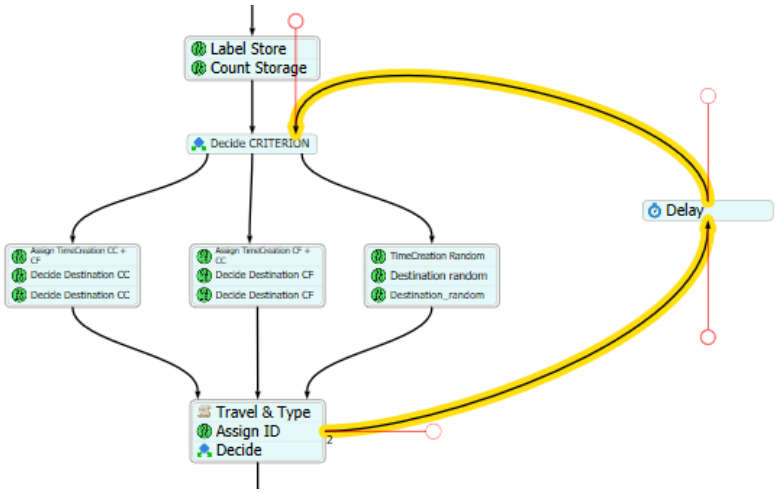

Figura 26 Process Flow: loop magazzino pieno

Un delay trattiene il token per un tempo di 50 secondi, tempo ciclo medio per un retrieval, prima di mandarlo nuovamente al punto della scelta del criterio e della futura destinazione.
Il funzionamento generale è lo stesso per i tre criteri, i quali si differenziano per il modo in cui riempiono il magazzino.

2.6.1 Criterio Closest Channel

Il primo criterio implementato nel modello è il *Closest Channel.* Come suggerisce il nome, si propone di riempire completamente un unico canale in tutti i livelli prima di passare al successivo, dal più vicino all'ascensore fino all'ultimo. Oltre alla scelta del canale, si è implementata anche la scelta del livello, scegliendo di riempire i livelli in ordine crescente di lontananza dal pavimento, ma questa logica sarà spiegata meglio nel prossimo criterio. Dalle figure, in cui abbiamo una singola tipologia

di item e nessun impedimento strutturale, si nota come il riempimento sia verticale, si riempiono in ordine il primo canale, il secondo e così via. Inoltre, si riempie prima il canale del terzo livello, più vicino nel modello alle baie di storage e retrieval posizionate sul pavimento.

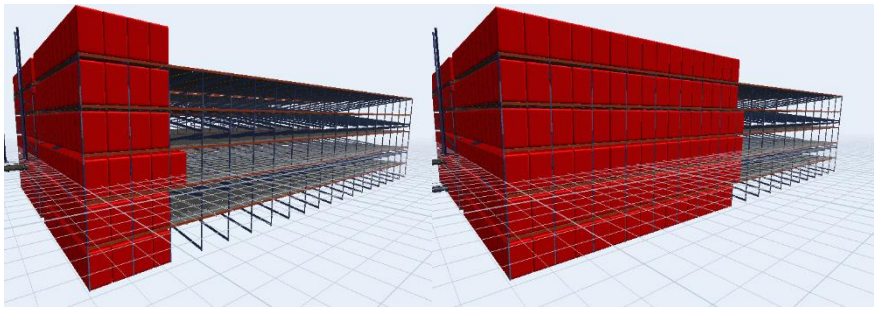

Figura 27 Modello con riempimento Closest Channel

Lo script per questa scelta, contenuto nel *Decide Destination CC*, usa strutture molto semplici. Per ogni rack, c'è un doppio ciclo *for*, il primo scorre i canali ed il secondo i livelli. Alla prima combinazione utile, livello-canale, in cui è possibile lo storage, la scelta è fatta ed il codice esce da entrambi i cicli. Come detto, si cicla contemporaneamente su entrambi le scaffalature, quindi può capitare che lo storage sia possibile sia a destra che a sinistra: in questi casi si sceglie casualmente tra i due. È utile ricordare che la destinazione è memorizzata in un array, formato da tre elementi, nell'ordine numero del livello, numero del canale e numero del rack, "1" per la scaffalatura di sinistra e "2" per quella di destra.

La vista dall'altro mostra il riempimento con *Closest Channel* del magazzino, in cui è rispettata la regola di stoccare item dello stesso tipo vicini.

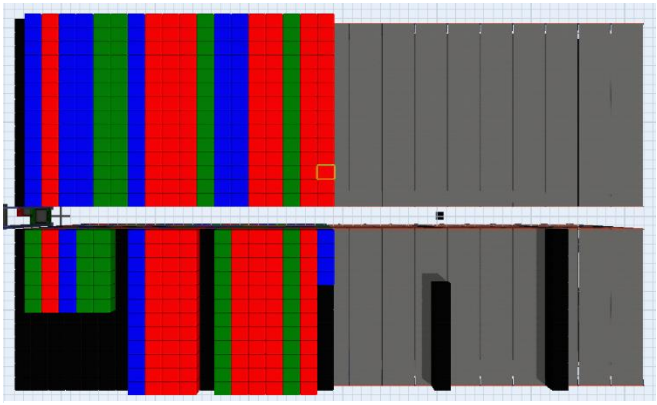

Figura 28 Vista dall'alto con criterio closest channel

2.6.2 Criterio Closest Floor

Il secondo criterio è il *Closest Floor*, in cui si preferisce riempire un intero livello prima di passare ad un altro. Come accennato nel precedente criterio, è stata implementata una scelta del livello intelligente, in cui si tiene conto della distanza dal pavimento, dove i veicoli, trasportati dall'ascensore, prelevano l'item da stoccare. I livelli nel modello, di default, sono numerati dal basso verso l'alto. Per questa ragione nello script principale è stato inserito un algoritmo di ordine che, confrontando le distanze sull'asse z (asse di movimento dell'ascensore), salva nella successione esatta i livelli in un array, denominato *order_level.* Nello specifico del caso modellato, l'ordine di riempimento dei livelli deve essere: 3-4-2-5-1-6, poiché il livello 3 è interrato solo per il 30% e dunque il più vicino al pavimento. È importante sottolineare, però, che tale array *order_level* è generato e aggiornato nello script generale, lanciato prima di ogni simulazione, e dunque si modificherà nel caso in cui l'utente cambi degli input, come il numero di livelli, la presenza di uno o più livelli sotterranei e la relativa percentuale di interramento. Anche in questo criterio la logica di scelta è stata ampliata con la scelta del canale più vicino all'ascensore, creando un metodo ibrido tra *Closest Floor* e *Closest Channel.* La logica è chiaramente visibile in figura: il primo a riempirsi è il terzo livello, dal primo all'ultimo canale. In successione, considerata la parte di magazzino sotterraneo, si alterneranno i vari livelli, fino alla massima capacità.

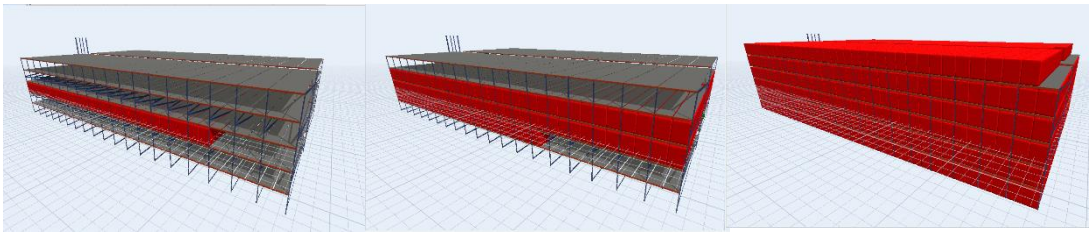

Figura 29 Modello con riempimento Closest Floor

Anche in questo caso, lo script utilizza due cicli *for*, con l'unica eccezione di iniziare con il ciclo dei livelli, riferiti all'array *order_level.* Al primo posto disponibili, verificata la reale presenza di un posto e la tipologia di item nel caso già stoccati nel determinato canale, si esce dal ciclo e si assegnano i valori all'array di destinazione. Dalla due viste laterali, interno ed esterno di un rack, è possibile osservare come proceda il riempimento nel caso di diverse tipologie di item e la corrispondenza dei colori tra le due viste: in un canale possono essere stoccati solo tipi uguali di item.

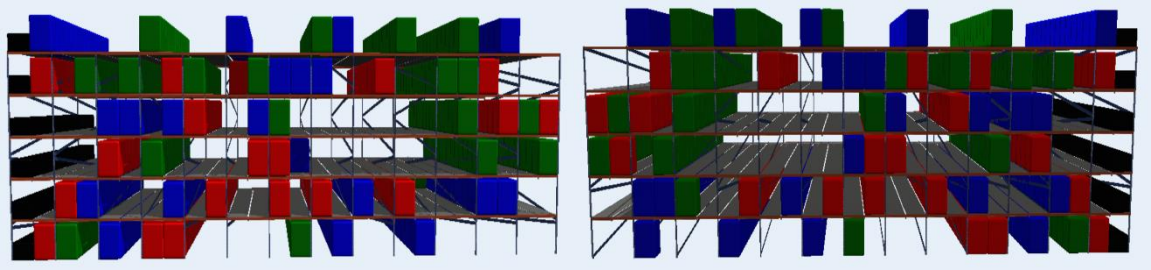

Figura 30 Vista laterale con corrispondenza tipologie

2.6.3 Criterio Random

Il terzo criterio implementato nel modello è di tipo randomico, dunque la scelta della destinazione, in termini di livello, canale e rack è casuale. Nella logica di scelta, però, si tiene conto dei canali in cui si è già stoccato qualche item, logicamente della stessa tipologia, ma non è stato completamente riempito, con lo scopo di saturare la capacità di questi stessi canali. Nella pratica, sia dopo la condizione iniziale, sia durante il normale esercizio, potrebbero verificarsi due situazioni:

1. C'è un canale per ogni rack, dove si è già stoccato un item di questa tipologia, su cui ricadrà la logica decisionale ed al massimo si sceglierà casualmente in quale dei due depositare l'item;
2. Non ci sono canali riempiti parzialmente di questa tipologia di item, dunque si sceglierà casualmente tra gli altri canali vuoti.

Queste diverse esigenze si traducono in un codice con logiche leggermente diverse dai precedenti criteri. Lo script è diviso in due parti distinte: nella prima i due cicli *for,* sui canali e sui i livelli, cercano i 2 canali in cui potrebbero esserci posti liberi per quel determinato item. In caso di esito negativo, si ripetono i due cicli *for* e si provvede a memorizzare in degli array le possibili destinazioni dei canali vuoti. Alla fine, dopo una scansione di tutto il magazzino, ci sarà un numero di possibili combinazioni, salvata in una variabile denominata *num_canali_disponibili,* la quale è aggiornata ogni volta che si identifica una possibile destinazione. Infine, casualmente, si sceglie un valore compreso tra 1 ed il numero di canali disponibili, che identificherà una allocazione univoca da memorizzare nell'array di destinazione. In figura è possibile osservare sia come la scelta sia ricaduta su un canale con la stessa tipologia ed ancora non pieno, sia il movimento dei veicoli: lo shuttle si ferma all'ingresso del canale, il satellite si stacca dallo shuttle e deposita l'item nella baia.

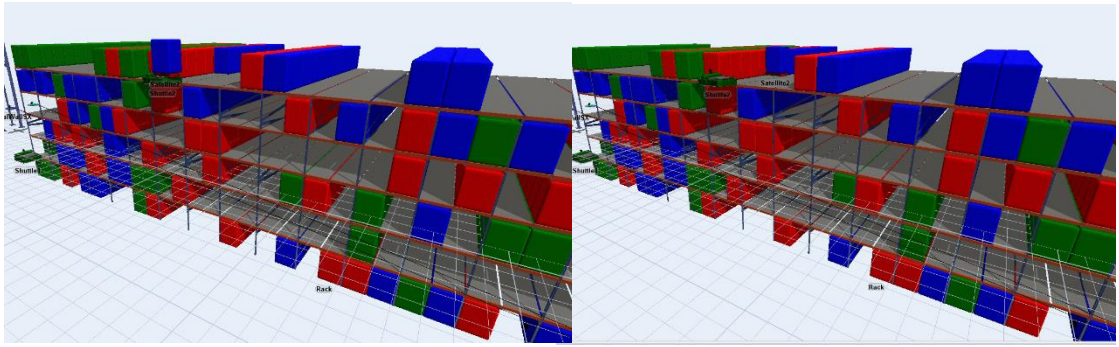

Figura 31 Modello con riempimento random

2.7 Retrieval Management

Il Process Flow presenta due rami paralleli, uno per lo storage ed uno per il retrieval. Le due parti sono, nella impostazione iniziale, molto simili.

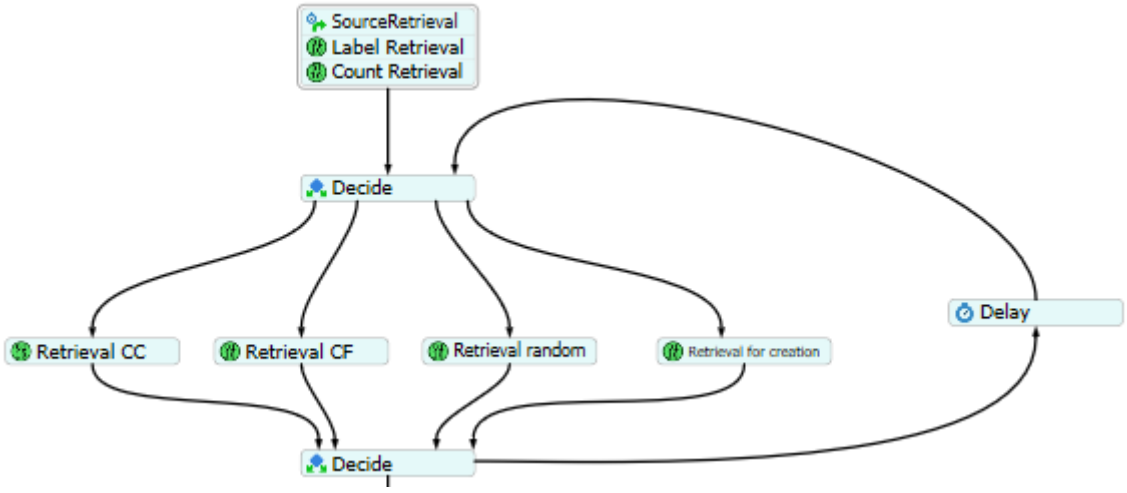

Figura 32 Process Flow: Retrieval management

Anche per il retrieval, la prima attività è una *Source*, cioè la creazione delle richieste. Analogamente alla sorgente dello storage, si è impostato un *inter arrival*, in cui l'utente può decidere la media e la varianza delle richieste di prelievo dal magazzino. La grande differenza è che, mentre per lo storage la creazione del token è subito legata alla creazione di un item corrispondente da stoccare, per il retrieval si ha la creazione solo di un token, in cui è immagazzinata l'informazione della tipologia di item da prelevare. Il *decide* successivo serve a definire, attraverso un valore numerico impostato dall'utente, il criterio di retrieval desiderato:

1. *Criterio Closest Channel;*
2. *Criterio Closest Floor;*
3. *Criterio Random;*
4. *Criterio for Creation.*

È chiaro come i primi tre metodi richiamino gli stessi visti per lo storage, mentre l'ultimo rappresenta una novità e saranno spiegati successivamente nel dettaglio. Alla fine della decisione, in ogni token sarà memorizzato un array di tre valori, indicanti livello, canale e rack in cui effettuare il ritiro dell'item. Esiste una logica base, comune a tutti i criteri: una volta delineato un canale in cui effettuare un retrieval, quel canale deve essere svuotato completamente. Per implementare questa decisione, dallo script generale, si genera una matrice che tiene conto del numero di tipologie diverse e per ognuna inserisce 4 colonne, "level", "channel", "Rack" ed "Item_prelev".

	Level	Channel	Rack	Item_prelev
Type 1	5	6	1	10
Type 2	3	23	2	4
Type 3	1	15	1	8

Figura 33 Gestione tabella retrieval per tipologia

Ad inizio simulazione tale matrice è inizializzata con valori tutti pari a 0. Per ogni tipologia, deciso il canale per il primo retrieval, si aggiorna la riga corrispondente con i valori di destinazione decisi dal criterio. Attraverso un comando di Flexsim, si contano gli item presenti nel canale e si memorizza tale informazione nell'ultima colonna. Al prossimo retrieval di questa tipologia, ci sarà un controllo su gli item prelevabili, cioè si verifica che il valore della colonna "Item_prelev" sia maggiore di zero, ed in questo caso la destinazione del ritiro sarà già decisa. Logicamente si riduce di una cifra gli item al momento prelevabili. Qualora il valore di item prelevabili fosse nuovamente 0, la decisione sarà effettuata tramite le logiche del criterio prescelto. Altra considerazione generale, simile a quella fatta per o storage, è sul problema di confrontare la situazione in cui si effettua la decisione e quella che ci sarà quando il ritiro dell'item accadrà realmente. La soluzione implementata è analoga a quella per lo storage, con delle matrici generali che inglobano le informazioni sui retrieval già assegnati. È chiaro come il sistema faccia una doppia verifica ogni qualvolta assegni un processo di storage: si può effettuare uno storage solo nei canali in cui non è previsto già nessun retrieval. Anche nel ramo del retrieval è presente un loop, corrispondente alla situazione in cui il magazzino sia completamente vuoto. In questo caso le richieste di prelievo di item non possono essere soddisfatte, dunque si rinviano i token, dopo un delay pari al tempo ciclo di uno storage, al criterio di scelta.

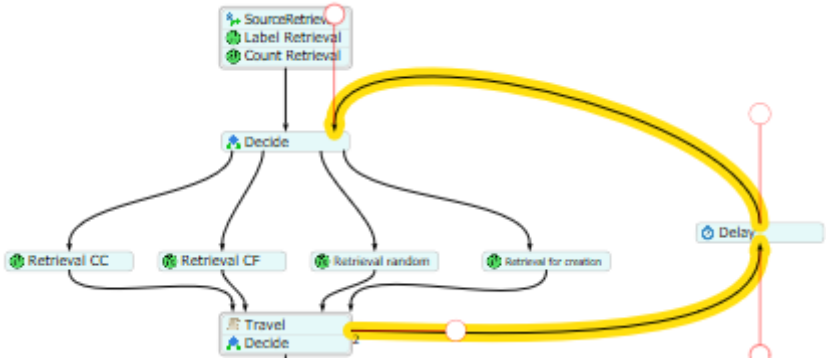

Figura 34 Process Flow: loop magazzino vuoto

2.7.1. Criterio Closest Channel, Closest Floor e Criterio Random

Questi tre criteri hanno logiche decisionali analoghe a quelle dello storage, logicamente tradotte per operazioni di ritiro degli item dal magazzino. Il criterio *closest channel,* attraverso due cicli *for,* effettua delle verifiche dal primo all'ultimo canale di ogni livello. Anche in queste logiche si è implementato un criterio ibrido, poiché a parità di canale si da precedenza ai livelli più vicini al piano terra. La verifica per ogni baia controlla la presenza di almeno un item e la sua tipologia, per assicurarsi di soddisfare la richiesta ricevuta. Al primo canale, conforme ai requisiti, identificato il doppio ciclo si interrompe e si salvano le informazioni nel solito array di destinazione. In figura si nota chiaramente come, partendo da una situazione di riempimento iniziale del 100%, si svuoti il canale più vicino all'ascensore ed in particolare nel livello più vicino al pavimento nella costruzione di questo modello.

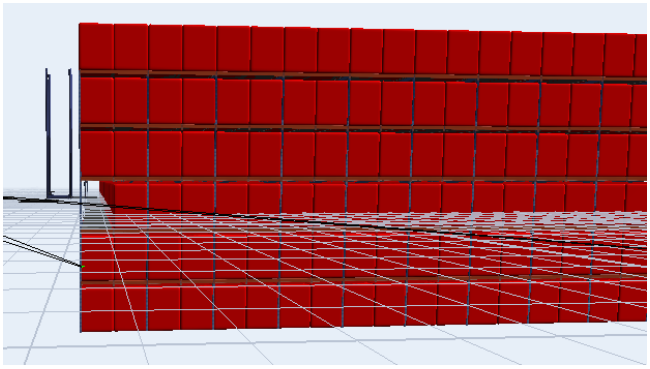

Figura 35 Vista laterale modello con retrieval Closest Channel

Una vista dall'alto sottolinea che, terminati gli item nel primo canale, la logica di simulazione svuota il secondo canale, in ogni livello.

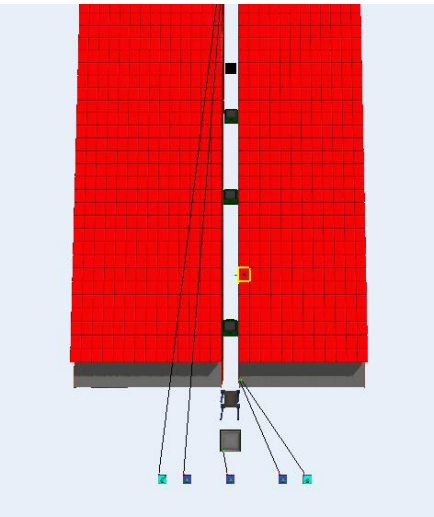

Figura 36 Vista dall'alto modello con retrieval Closest Channel

Logica identica è implementata nel criterio *Closest Floor*, il quale si differenzia dal precedente solo per l'ordine dei due cicli *for.* Infatti, in questo caso, la prima scelta è del livello, attraverso l'array *order_level* che ordina i livelli in base alla distanza dal piano terra. Scelto il livello, si scorre su tutti i canali, dal primo all'ultimo. Identificato il primo canale utili, verificando la presenza di almeno un'item e la tipologia, si interrompe il ciclo e si salva la destinazione trovata. In figura si vede come si svuoti gradualmente il terzo livello, prima di passare a quello successivo. Inoltre, si nota come i ritiri iniziali dal magazzino siano uguali per i primi due criteri, poiché entrambe le logiche svuotano prima i canali vicini all'ascensore.

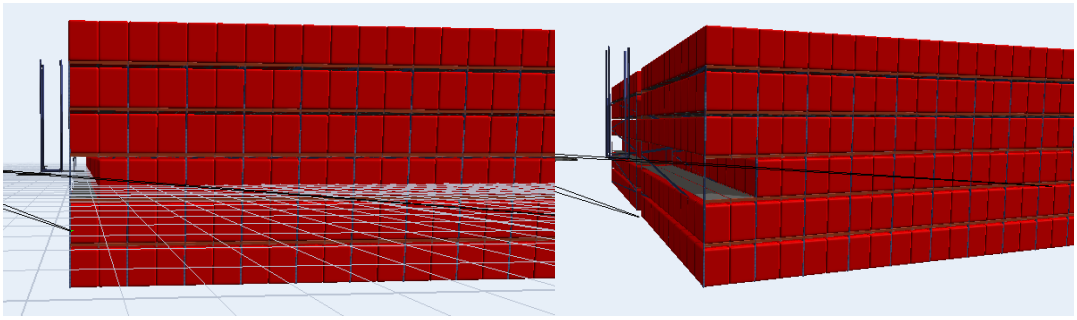

Figura 37 Vista laterale modello con retrieval Closest Floor

Anche il *criterio random* è simile all'analogo per lo storage. Attraverso i due cicli *for*, si scannerizza tutto il magazzino, memorizzando in degli array canale, livello e rack per ogni baia che soddisfa i requisiti della richiesta e tenendo traccia del numero di possibili soluzioni in una variabile locale. Alla fine, si sceglie casualmente un numero compreso tra 1 ed il numero di canali disponibili per il retrieval e si salva nell'array di destinazione la combinazione scelta.

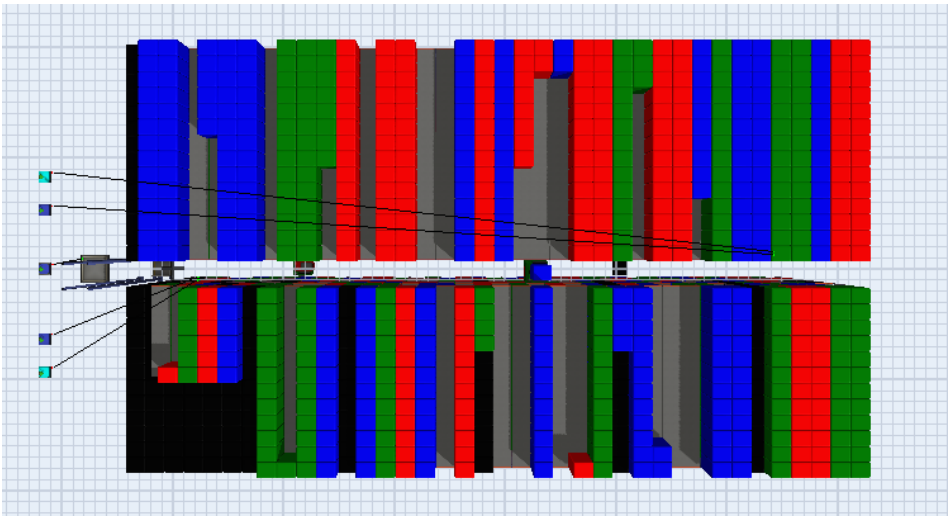

Figura 38 Vista dall'alto modello con retrieval random e più tipologie di item

2.7.2 Criterio For Creation

Per il retrieval è stato inserito un ulteriore criterio che tiene conto della creazione degli item stoccati. Infatti, soprattutto per alcune applicazioni come il campo alimentare o farmaceutico, le materie immagazzinate sono soggette a delle scadenze e ad un deperimento legato al passar del tempo. È importante aver traccia delle informazioni di tutto ciò che è immagazzinato per poter risalire all'item con il tempo di creazione più piccolo, cioè il più vecchio. Lo script decisionale è molto semplice e introduce piccole modifiche al codice del criterio random, poiché rimane la scannerizzazione di tutto il magazzino, introducendo, però, un ulteriore controllo. Infatti, per ogni canale, si controlla il tempo di creazione di ogni item stoccato, aggiornando la baia di ritiro ogni volta che si incontra un'item con età maggiore. Alla fine dei cicli, sia per i canali che per i livelli, nell'array di destinazione saranno indicate le coordinate del canale in cui iniziare le pratiche di retrieval.

ProcessFlow - RetrievedItems Entries

value	ID	Type	Rack	Level	Channel	Time Creatior	Retrieval Sta	Retreival Cyc	Stored Time	Energy Shutt	Energy Satell	Energy Lift
id:1102 instance: ProcessFlow	12	1	Sx	6	7	110.15	1527.14	59.18	19.27	20091.81	4600.94	0
id:1118 instance: ProcessFlow	11	1	Sx	6	7	100.15	1652.49	59.92	264.54	46791.09	12569.02	23077.07
id:1250 instance: ProcessFlow	10	1	Sx	6	7	90.15	1779.18	60.56	512.35	46791.09	14367.54	23077.07
id:1213 instance: ProcessFlow	9	1	Sx	6	7	80.15	1907.11	61.16	762.60	46791.09	16225.72	23077.07
id:1182 instance: ProcessFlow	8	1	Sx	6	7	70.15	2036.24	61.76	1015.24	46791.09	18256.27	23077.07
id:1141 instance: ProcessFlow	7	1	Sx	6	7	60.15	2166.56	62.36	1270.28	46791.09	20461.57	23077.07
id:1113 instance: ProcessFlow	6	1	Sx	6	7	50.15	2298.08	62.96	1527.71	46791.09	21754.71	23077.07
id:1284 instance: ProcessFlow	5	1	Sx	6	7	40.15	2430.80	63.56	1787.53	46791.09	22004.67	23077.07
id:1196 instance: ProcessFlow	4	1	Sx	6	7	30.15	2564.72	64.16	2049.75	46791.09	22254.67	23077.07
id:1162 instance: ProcessFlow	3	1	Sx	6	7	20.15	2699.83	64.76	2314.37	46791.09	22504.70	23077.07
id:1240 instance: ProcessFlow	2	1	Sx	6	7	10.15	2836.15	65.36	2581.38	46791.09	22754.74	23077.07
id:1277 instance: ProcessFlow	1	1	Sx	6	7	0.15	2973.66	65.96	2850.80	46791.09	23004.81	23077.07

Figura 39 Informazioni item stoccati in magazzino

In figura è presente una tabella dell'ambiente Flexsim, in cui sono memorizzati tutti gli item prelevati dal magazzino e le loro informazioni. È facile testare che le logiche decisionali svuotino interamente il canale, prelevando nell'ordine item con un "Time Creation" decrescente, cioè dal più giovane al più vecchio, che logicamente è stato stoccato prima nel magazzino e si trova dietro altri item nel canale.

2.8 Scelta veicolo

La scelta della coppia shuttle/satellite da impiegare per un'operazione di storage o retrieval si basa sulla distanza, cioè sul calcolo dei veicoli più vicini. Nel caso dello *stoccaggio,* la missione inizia con il ritiro dell'item nella baia di storage posta sul pavimento, con valore di asse z pari a 0. Lo shuttle, al momento della presa in carica dell'ordine, si troverà in un determinato livello, dinanzi ad un determinato canale, dunque nella determinazione della distanza si sommerà anche il valore relativo all'asse x, asse di movimento degli shuttle lungo i livelli. Invece, nel caso di retrieval di un item, c' è da tenere in conto anche la posizione dell'item nel canale di ripresa, cioè c'è da considerare un ulteriore addendo riferito all'asse y, asse di movimento dei satelliti. Questo fattore si aggiungerà in valore assoluto, essendo l'origine dell'asse al centro e, quindi, con valori negativi nei canali del rack sinistro. La scelta, in entrambi i rami principali del process flow, è implementata in un *custom code*, denominato "*Distance Sh&Sat"*.

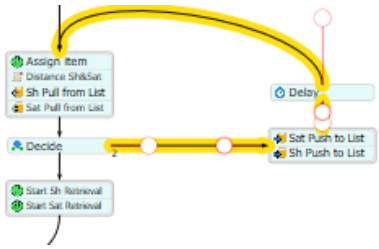

Figura 40 Process Flow: loop del controllo posizione shuttle per storage

Come si nota in figura, le attività successive sono l'acquisizione dei due veicoli da due specifiche liste e, poiché la frequenza delle richieste spesso è maggiore del tempo ciclo per asservire un processo, sia di storage che di retrieval, qui i token restano in coda in attesa che i veicoli siano disponibili. Inoltre, in questo punto si incontra un ulteriore *decide,* che introduce il secondo loop decisionale del sistema. Infatti, per motivi di sicurezza, in ogni livello non può esserci più di una coppia di veicoli

contemporaneamente. Dunque, una volta assegnati i veicoli di competenza per il determinato ordine, ma soprattutto dopo averli "prenotati" per completare uno stoccaggio o una ripresa di un item, è necessario verificare che nel livello di destinazione non sia presente già un ulteriore shuttle. In caso di esito negativo, il token è inviato in un loop, nel quale libera i due veicoli impegnati, aspetta un ritardo temporale pari ad il tempo medio di un ciclo, e ritorna nel punto in cui si scelgono i veicoli più vicini.

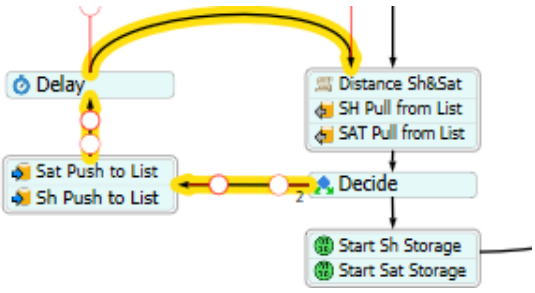

Figura 41Process Flow: loop del controllo posizione shuttle per retrieval

2.9 Logiche di acquisizione dell'ascensore e parallelizzazione attività

La zona centrale del process flow è dedicata alle logiche di acquisizione e rilascio dell'ascensore. Mentre il numero dei veicoli, shuttle e satelliti, può essere pari al numero dei livelli, l'ascensore nel modello di magazzino, così come è stato creato nella simulazione, è singolo. È intuibile come sia necessario sfruttare al massimo il suo utilizzo, evitando perdite di tempo o attese inutili. Nel software Flexsim, l'ascensore è un *task executer,* denominato *lift*; nel process flow è visualizzato, invece, come una risorsa ed esistono due attività nella libreria implementata nel software, *"Acquire resource"* e *"Release resource"*. Quando un token passa per queste azioni, prenota le prestazioni della risorsa fino al rilascio della stessa, rendendolo di conseguenza indisponibile per altre operazioni. Per enfatizzare tale logica si è posto l'*acquire* al centro del process flow.

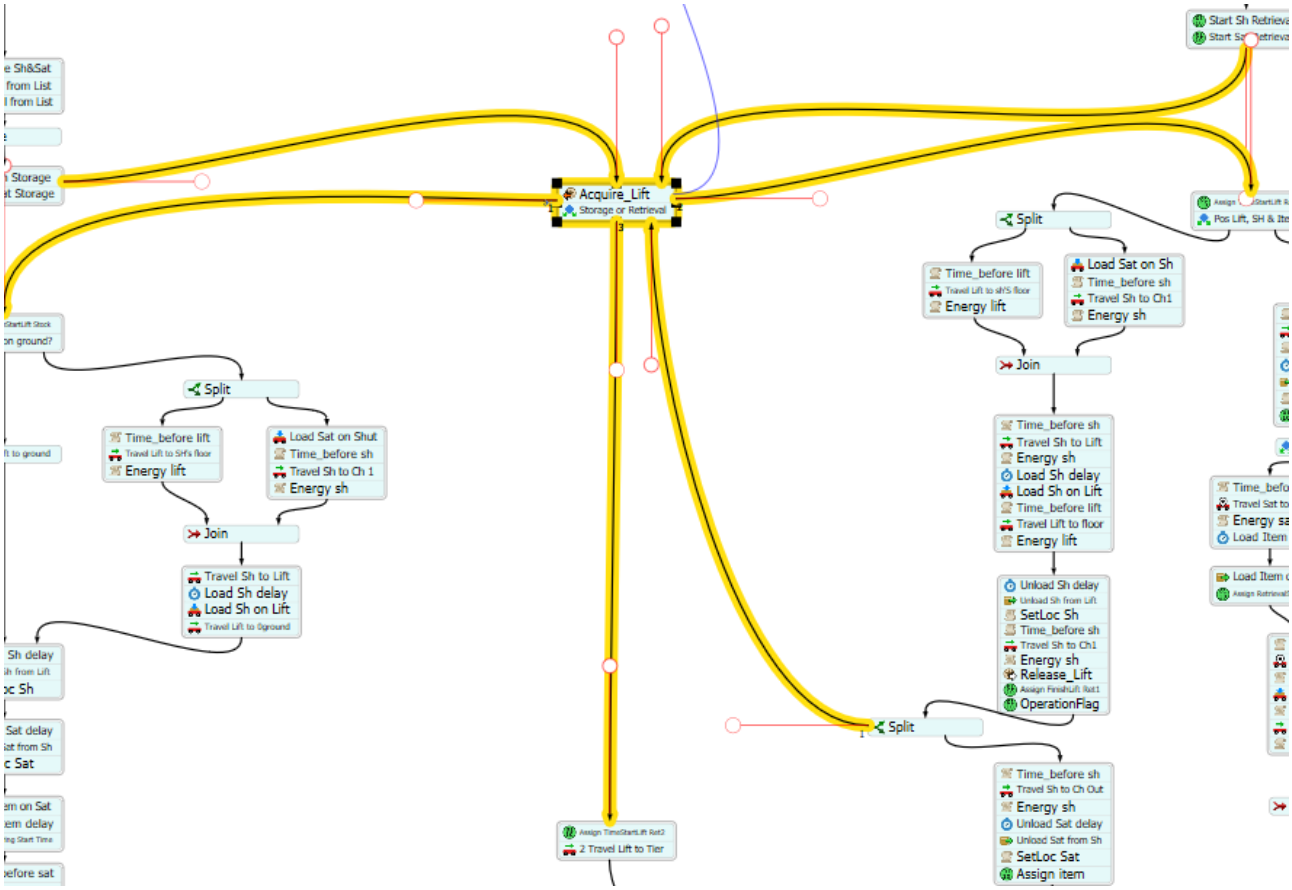

Figura 42 Process Flow: acquisizione ascensore

In figura è visibile come siano presenti tre diversi situazioni da tenere in considerazione nell'ottica generale della simulazione. Il *decide* posto dopo l'acquisizione, legge un'informazione scritta nel token, per indirizzarlo al ramo in uscita corrispondente quando arriva il suo turno.

2.9.1 Attività storage

Nel processo di storage, il token richiede l'acquisizione dell'ascensore dopo aver deciso la destinazione ed aver prelevato dalla lista lo shuttle ed il satellite. Come ormai chiaro, nel processo di storage, l'item deve essere prelevato dalla baia posta sul pavimento e possono presentarsi tre situazioni differenti, come in figura.

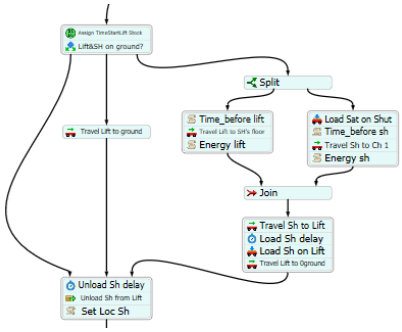

Figura 43 Process Flow: comportamento storage

1. Ascensore e veicoli si trovano già al livello 0, cioè ad altezza pavimento, dopo aver asservito un'operazione di retrieval e quindi il prelievo dalla baia dello storage è immediato;
2. Lo shuttle è sul pavimento, in prossimità della baia di deposito degli item presi dal magazzino, ma l'ascensore è su un altro livello. In questo caso il satellite può prelevare direttamente l'item da stoccare, riaccoppiarsi con lo shuttle ed attendere l'arrivo dell'ascensore che li porterà al livello di destinazione dell'item;
3. Sia shuttle che ascensore sono allocati nei diversi livelli del magazzino. È necessario, quindi, che l'ascensore si muova verso il livello in cui si trova lo shuttle, per accoppiarsi e dirigersi verso il pavimento. Qui entra in gioco una funzione molto utile per parallelizzare le azioni, lo *SPLIT*. Questa attività divide il token originario in due copie identiche, contenenti le stesse informazioni. In questo caso specifico si vuole separare i movimenti dell'ascensore e dello shuttle, per diminuire i tempi morti. L'ascensore si dirigerà verso il livello dello shuttle, mentre quest'ultimo si dirigerà verso il primo canale, in prossimità della zona di accoppiamento con il lift. Solo al concludersi delle due azioni, attraverso l'attività *JOIN,* i due token gemelli si riuniranno per riformare quello originario ed il flusso delle attività potrà riprendere.

Il proseguo delle attività per lo storage è molto lineare, arrivati al livello di destinazione, i veicoli e l'item scendono dall'ascensore, rilasciando la risorsa per la prossima operazione. La coppia shuttle/satellite raggiungere il canale destinato a ricevere l'item e concluderanno le operazioni di scarico.

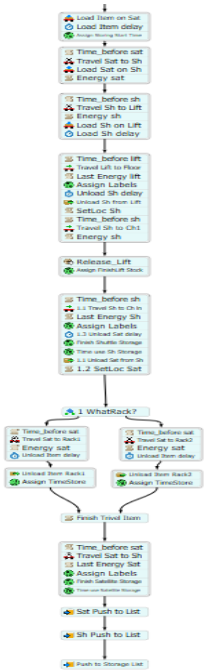

Figura 44 Process Flow: fine processo storage

Ultima biforcazione del flusso è in corrispondenza del *decide "What's Rack?"* e si ritrova uguale anche nel ramo del retrieval. Tale domanda nasce dalla configurazione inziale delle due scaffalature: dallo script sono costruite sovrapposte e successivamente quella di sinistra è ruotata di 180°. Ciò comporta che la numerazione dei canali è contrapposta, mentre il canale più vicino all'ascensore a destra è identificato come "channel 1", nel rack di sinistra il canale numero 1 è il più lontano. Il decide, naturalmente, legge l'informazione del rack nell'array di destinazione, salvato nel token, e configura adeguatamente le coordinate di invio dei veicoli.

2.9.2 Attività retrieval

Nel ramo dedicato alle operazioni di ripresa di un item dal magazzino, dopo acquisizione dell'ascensore, si devono valutare le posizioni dell'item da ritirare e dei veicoli assegnati al processo. In realtà si dovrebbe prendere in considerazione anche il livello in cui si trova l'ascensore, ma tale valutazione si trascura poiché, nel caso il lift fosse già al livello giusto, il process flow gli assegnerebbe un viaggio nullo, senza dispendio di energia e tempo ma permettendo un alleggerimento grafico del flow stesso.

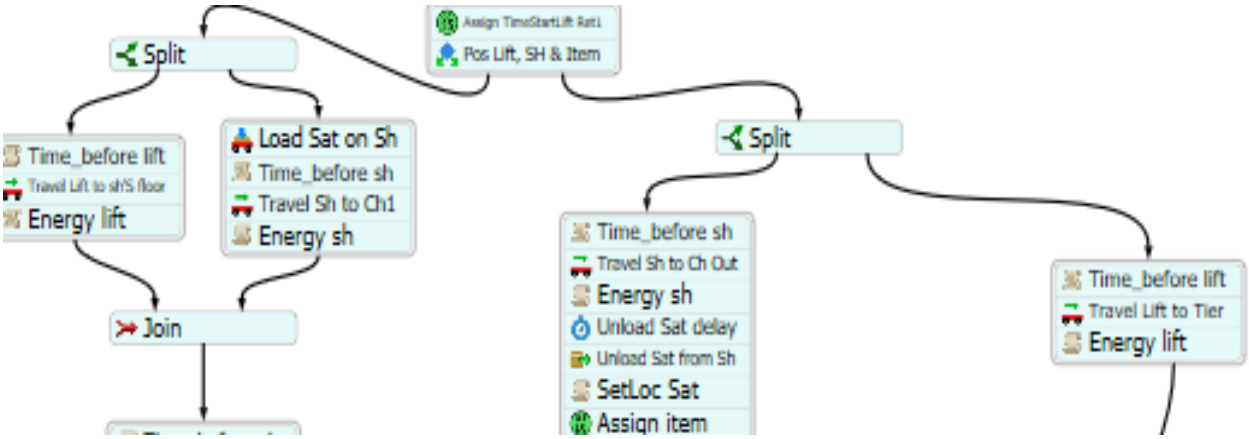

Figura 45 Process Flow: comportamento retrieval

Dunque, le situazioni possibili sono due:

- Nel caso peggiore, shuttle e satellite sono allocati in un piano diverso da quello dell'item da ritirare, quindi hanno bisogno dell'intervento dell'ascensore per raggiungere l'altezza necessaria. La prima parallelizzazione, sempre attraverso uno *split*, divide il movimento dell'ascensore fino al livello dei veicoli ed il movimento di quest'ultimi verso il primo canale, quello più vicino al lift. Al termine di queste due attività, i tre *task executer* si accoppiano e l'ascensore trasporta lo shuttle al canale del ritiro. Adesso, per permettere all'ascensore magari di asservire un'altra operazione mentre shuttle e satellite raggiungono l'item da ritirare, la risorsa lift è rilasciata.

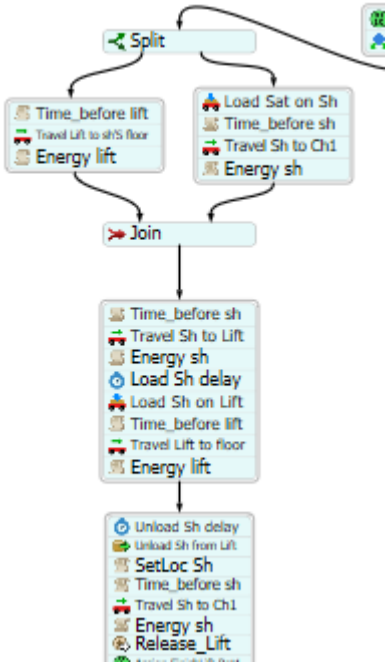

Figura 46 Process Flow: dettaglio parallelizzazioni attività retrieval

Il token, allora, è diviso nuovamente: una copia aspetta la verifica degli ordini di prenotazione dell'ascensore, per riprenotarne la disponibilità, e l'altra copia esegue le attività di retrieval, fino al raggiungimento dello shuttle, con su caricato il satellite che a sua volta trasporta l'item, del primo canale, in attesa dell'ascensore.

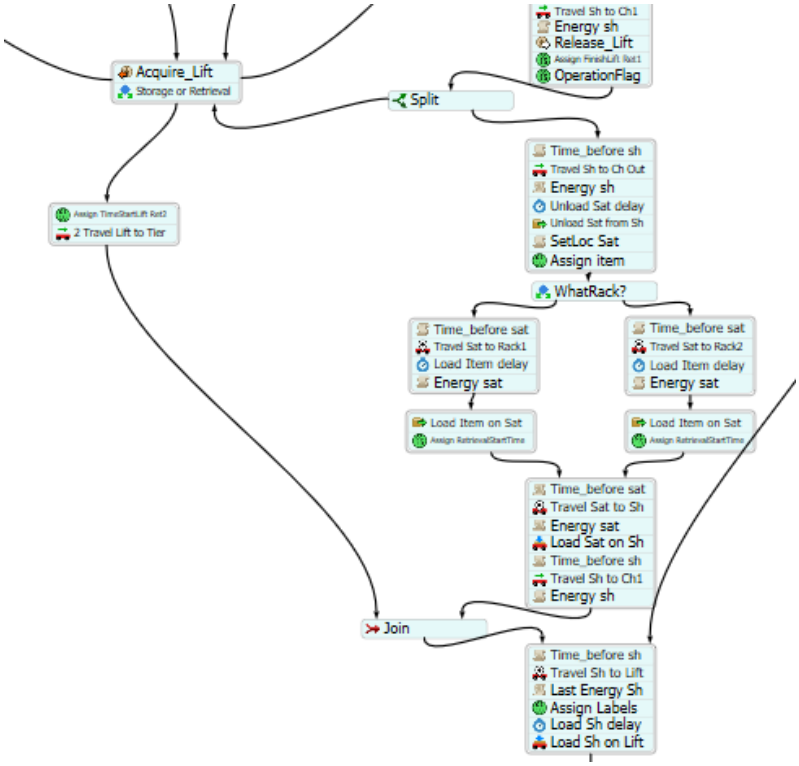

Figura 47 Process Flow: parallelizzazione attività e gestione ascensore

- I veicoli, assegnati per il ritiro, si trovano sullo stesso livello dell'item da prelevare. In questo caso il token è diviso attraverso uno *split,* parallelizzando le attività di ritiro effettuate da shuttle e satellite, ed il raggiungimento del livello da parte dell'ascensore. Come anticipato, nel caso l'ascensore fosse già in prossimità di quel livello, il viaggio sarebbe nullo e non comprometterebbe i risultati della simulazione.

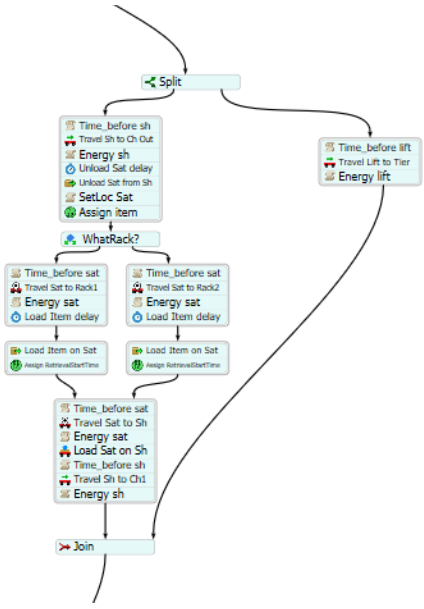

Figura 48 Processo Flow: dettaglio gestione retrieval

Anche nelle operazioni di retrieval, analogamente a quanto accade per gli storage, dinanzi al canale indicato nell'array di destinazione, il sistema si chiede in quale dei due rack è situato l'item da prelevare, per poter così settare le coordinate di movimento del satellite. Da questo punto, il process flow torna ad essere unico e le ultime operazioni sono molto lineari: l'ascensore trasporta i veicoli e l'item sul pavimento e l'item prelevato è depositato nella baia dei retrieval.

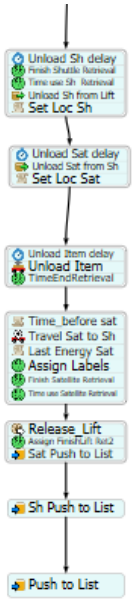

Figura 49 Process Flow: fine processo retrieval

2.10 GUI (Graphic User Interface)

Le interfacce utente sono uno strumento molto potente dell'ambiente Flexsim, in cui è possibile editare degli elementi grafici interessanti e molto efficienti. Nello specifico, l'obiettivo è stato duplice:

1. Rendere agevole l'utilizzo del simulatore anche a persone non esperte, che in particolare non conoscono il software e le sue dinamiche;
2. Semplificare agli utenti la personalizzazione del modello, avendo a disposizione la possibilità di cambiare velocemente i parametri interessati, senza dover ricercare tra le molte variabili del software, rischiando di inquinare il modello stesso.

La GUI (Graphic User Interface) implementate è formata oltre che da elementi grafici, come il logo del Politecnico di Torino, il logo della Flexcon, azienda distributrice in Italia del programma Flexsim,

da tre ambienti fondamentali, i quali guidano passo dopo passo l'utente dalla costruzione del modello all'avvio della simulazione.
Nella pagina *Warehouse Structure* si procede all'assegnazione dei parametri strutturali del modello:

- *Number of Levels*, il numero desiderato dei livelli;
- *Number of Channels,* il numero desiderato dei canali;
- *First Under Level,* specificare se una parte del magazzino è posizionato sotto il livello del pavimento. Bisogna indicare il numero del primo livello interrato, cioè quello più vicino al pavimento, ricordando che la numerazione dei livelli è crescente dal basso verso l'altro.
- *% First Under Level,* cioè di quale percentuale il primo livello interrato è sotto il pavimento. È un valore tra 0 ed 1.
- *Capacity Channels Left*, capacità dei canali del rack sinistro.
- *Capacity Channels Right,* capacità dei canali del rack destro. Tali valori sono le capacità massime, modificabili successivamente con un file Excel.

Il magazzino è pronto per essere costruito, attraverso il comando *Build the Warehouse*. Nell'ambiente grafico si realizzerà il modello con le dimensioni inserite, visualizzabile cliccando su *3D Model Wiew.*

Figura 50 GUI: prima schermata

Nel secondo ambiente, *Vehicle Parameters*, si può modificare:

- *Number of Vehicle*, numero dei veicoli, riferito agli shuttle ed ai satelliti in quanto l'ascensore è sempre singolo;
- *Acceleration* e *Speed*, velocità ed accelerazione specifica sia per shuttle e satellite che per l'ascensore.

Di default il modello ha inseriti valori reali, presi da dei costruttori di tali veicoli ed utilizzati per la raccolta dati di questo lavoro di tesi. Quindi, in mancanza di tali valori, un utente può utilizzare tali informazioni per una prima analisi.

Figura 51 GUI: seconda schermata

L'ultimo ambiente, *Simulation Parameters,* è il fondamentale per l'avvio della simulazione. Inizialmente è possibile modificare i parametri del sistema:

- *Fill Rate Rack Dx* e *Fill Rate Rack Sx*, cioè impostare il tasso di riempimento iniziale dei due rack, indicando la percentuale, da 1 a 100, della capacità totale;

- *Storage Rate Mean* e *Retrieval Rate Mean,* la media delle richieste di storage e retrieval. Con questi valori, attraverso una distribuzione uniforme, si genereranno dei token che attraverseranno le specifiche attività di processo per effettuare le operazioni;
- *Storage Rate Variance* e *Storage Rate Mean,* cioè le variazioni per i due processi rispetto alla media assegnata;
- *Storage Criteria* e *Retrieval Criteria,* menù a tendina in cui scegliere il criterio selezionato. È importante sottolineare come, nel menù, compaiano solo i valori che è possibile selezionare e come sia indicati a fianco, in modo da evitare errori che possano portare al fallimento della simulazione.
- *Number of Items Type,* in cui specificare quante differenti tipologie di item si vuole inserire nel magazzino.

Figura 52 GUI: terza schermata

Inseriti tutti i parametri, è possibile avviare la simulazione, con semplici passaggi da effettuare in ordine:

- *Import Tables from Excel*, da utilizzare solo nel caso si vogliano inserire delle eccezioni alle capacità dei singoli canali nel magazzino, magari per la presenza di impedimenti strutturali o solo per verificare il funzionamento in caso di manutenzione in alcuni canali. In un particolare percorso, già impostato, è presente un file Excel, denominato *Capacity,* con due fogli: in ognuno c'è un riferimento al numero di canali e di livelli, in modo che ogni casella riconduca inequivocabilmente ad un determinato canale in cui scrivere la capacità voluta. Logicamente, come indicato dal nome dei fogli stessi del file *Capacity*, il primo è riferito al rack sinistro ed il secondo al rack destro.

A questo punto, attraverso la barra di comando, è possibile:

- *Run*, avviare la simulazione;
- *Stop,* stoppare la simulazione. È una pausa reversibile, in quanto con *Run* la simulazione riprenderà dal punto in cui è stata fermata.
- *Reset,* nel caso in cui si voglia resettare il modello dopo lo stop. Inserendo nuovamente i parametri, che comunque sono già salvati per un nuovo lancio, è possibile avviare una nuova simulazione.

3. Raccolta dati

Ultimato il modello e verificato il funzionamento di tutte le logiche assemblate, soprattutto nella loro interazione, è stata effettuata una raccolta dati accurata, sfruttando il Process Flow di Flexsim. La sua struttura e la sua capacità di prevedere al suo intero, sia in attività specifiche come i *custome code* o semplicemente nell'assegnazione di *labels* ai *token,* dei codici scritti in *FlexScript,* rende agevole la memorizzazione di alcuni istanti di tempo notevoli, in cui avviene una determinata azione.

3.1 Dati relativi allo Storage ed al Retrieval

Per le missioni di deposito e prelievo degli item in magazzino, le informazioni relative ai tempi delle varie attività sono memorizzate come informazioni legate al *token* e quando possibile allo stesso item, attraversi l'attività *assign labels.*

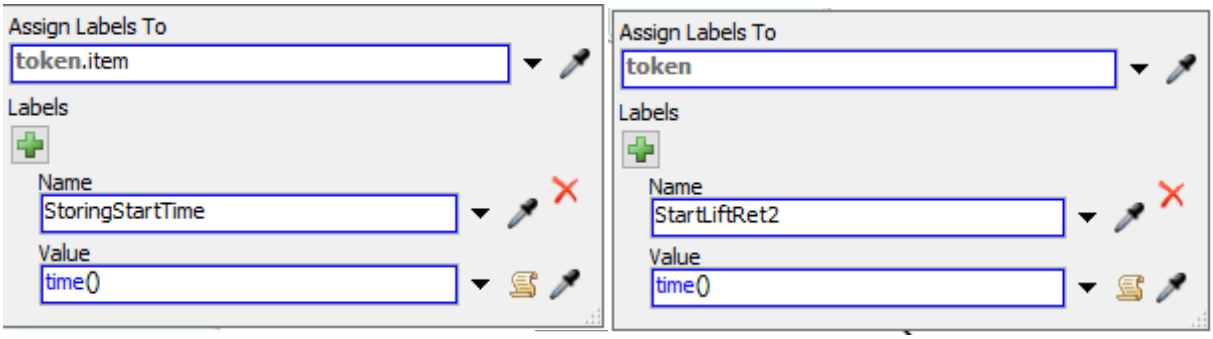

Figura 53 Process Flow: Assign labels

Al termine dei due rami principali del Process Flow, cioè alla conclusione di un'operazione di storage o di una ripresa dal magazzino, i token sono inviati in due liste, *StoredItems e RetrievedItem,* in cui dunque si avranno tutte le informazioni relative agli item stoccati in magazzino e quelli invece che hanno subito un processo di retrieval.

Figura 54 Process Flow: Push to List

Tali informazioni sono accessibili sotto forma di tabelle, in cui sono state calcolate le informazioni necessarie per ricostruire la storia temporale di ogni item e la sua locazione nel magazzino. Infatti, sia per gli storage che per i retrieval, i dati memorizzati sono:

- *Value,* che indica l'identificativo del token relativo al pallet;
- *ID,* valore numerico che identifica inequivocabilmente un pallet;
- *Type,* tipologia di item;
- *Time Creation,* istante in cui l'item, o meglio il corrispettivo token con la richiesta di operazione, è stato creato;
- *Rack, Level* e *Channel,* cioè la tripletta indicante la ubicazione nel magazzino in cui depositare o prelevare l'item;
- *Start Time,* istante in cui ha inizio il processo di storage o retrieval;
- *Cycle Time,* tempo ciclo dell'operazione;
- *Waiting Store,* attesa dell'item in coda prima di essere stoccato;
- *Stored Time,* tempo di permanenza in magazzino dell'item;
- *Vehicle,* veicolo utilizzato per l'operazione;
- *Energy shuttle, satellite e lift,* energia consumata dai veicoli nelle singole operazioni.

value	ID	Type	TimeCreation	Rack	Level	Channel	StoringStartTime	Storing Cycle Time	WaitingStore	Vehicle	Energy Shuttle	Energy Satellite	Energy Lift
id:1 instance: ProcessFlow	1318	3	0.15	Dx	3	38	50.93	51.75	50.78	2	42786.62	37910.42	0
id:1321 instance: ProcessFlow	1319	2	195.77	Dx	3	1	235.36	42.18	39.58	2	4348.44	18651.73	0
id:1328 instance: ProcessFlow	1320	3	396.43	Dx	3	38	448.83	51.15	52.40	3	4348.44	37660.58	0
id:1333 instance: ProcessFlow	1321	1	592.82	Dx	4	1	681.12	44.02	88.30	2	4348.44	18651.73	28016.14
id:1338 instance: ProcessFlow	1322	1	799.46	Dx	4	38	951.42	53.59	151.96	1	4348.44	37910.42	28016.14
id:1342 instance: ProcessFlow	1323	1	1005.00	Dx	4	1	1145.77	44.87	140.77	1	4348.44	20347.91	40770.31
id:1347 instance: ProcessFlow	1324	2	1212.59	Dx	3	1	1241.91	43.03	29.33	3	4348.44	20347.91	0
id:1352 instance: ProcessFlow	1325	3	1426.35	Dx	3	38	1534.59	50.55	108.25	3	4348.44	37410.79	0
id:1354 instance: ProcessFlow	1326	1	1620.84	Dx	4	38	1662.26	52.99	41.42	1	4348.44	37660.58	40926.84
id:1360 instance: ProcessFlow	1327	1	1807.18	Dx	4	1	1895.72	45.72	88.54	1	4348.44	22394.85	41448.61

Figura 55 Tabella con gli storage

value	ID	Type	Rack	Level	Channel	Time Creation	Retrieval Start Time	Retreival Cycle Time	Stored Time	Energy Shuttle	Energy Satellite	Energy Lift
id:1323 instance: ProcessFlow	808	2	Dx	4	14	-996	331.44	59.19	331.44	72557.48	22711.92	13432.47
id:1326 instance: ProcessFlow	805	2	Dx	4	14	-276	453.27	67.98	453.27	39015.31	22461.89	21356.64
id:1330 instance: ProcessFlow	678	3	Dx	2	17	-977	592.09	63.30	592.09	83859.52	22711.92	21356.64
id:1334 instance: ProcessFlow	674	3	Dx	2	17	-551	772.20	62.70	772.20	83401.29	22461.89	0
id:1337 instance: ProcessFlow	803	2	Dx	4	14	-634	867.70	57.99	867.70	53628.03	10086.49	12754.17
id:1340 instance: ProcessFlow	799	2	Dx	4	14	-77	1041.61	57.39	1041.61	68977.33	21961.91	21356.64
id:1344 instance: ProcessFlow	951	1	Dx	5	34	-995	1251.36	72.80	1251.36	56364.47	22711.92	22216.86
id:1349 instance: ProcessFlow	945	1	Dx	5	34	-106	1417.47	72.20	1417.47	132990.87	22461.89	22216.86
id:1356 instance: ProcessFlow	654	3	Dx	2	17	-757	1754.40	62.10	1754.40	86773.79	22211.88	0
id:1359 instance: ProcessFlow	650	3	Dx	2	17	-493	1879.30	98.04	1879.30	45243.58	21961.91	0

Figura 56 Tabella con i retrieval

Nello specifico è possibile identificare alcune caratteristiche. Nel retrieval gli item possono avere *time creation* minore di zero, condizione inserita per distinguere gli item inseriti nella condizione

iniziale e per testare la modalità di retrieval che tiene conto dell'anzianità dei pallet. Però, con questa condizione, lo *stored time* diventa un dato non indicativo della simulazione per questi item. È, però, un dato molto significativo per gli item che nel corso della simulazione subiscono entrambi i processi, identificabili dal proprio ID. In un ulteriore tabella è possibile leggere i parametri della simulazione lanciata e, dati più importanti, la quantità di Storage Retrieval sono giunti in input al sistema. In questo modo, rapportandoli con le operazioni realmente effettuate nel tempo di simulazione, banalmente il numero di righe delle tabelle precedenti, è deducibile la percentuale di risposta del sistema alle richieste in ingresso.

	Value
Fill Rate	19
Storage Rate Mean	200
Storage Rate Variance	15
Retrieval Rate Mean	200
Retrieval Rate Variance	100
Number Item Types	3
Storage Criteria	1
Retrieval Criteria	4
Number AGV	3
Storage Input	20
Retrieval Input	19

Figura 57 Tabella con parametri di simulazione

Un'ultima tabella calcola delle statistiche relative ai vari veicoli, ascensore, shuttle e satelliti, indicando per ognuno di essi:

- *Storing Time,* tempo totale impiegato in operazioni di storage;
- *% Time Storage,* percentuale del tempo totale di simulazione dedicato agli stoccaggi;
- *Retrieval Time,* tempo totale impiegato in operazioni di retrieval;
- *% Time Retrieval,* percentuale del tempo totale di simulazione dedicato alle riprese da magazzino;
- *Total time,* somma dello *Storing Time* e del *Retrieval Time;*
- *% Total Time,* percentuale di tempo di utilizzo di un veicolo rispetto al tempo totale di simulazione;
- *Energy Storage,* consumo totale di energia nelle operazioni di storage;
- *Energy Retrieval,* consumo totale di energia nelle operazioni di retrieval;
- *Total Energy,* consumo totale di energia, somma dei precedenti due addendi.

	Storing Time	% Time Storage	Retrieval Time	% Time Retrieval	Total Time	% Total Time	Energy Storage	Energy Retrieval	Total Energy
Lift	1284.45	34.35	2044.38	53.86	3328.84	87.70	318306.99	301242.01	619549.00
Shuttle 1	871.31	24.73	313.12	8.29	1184.43	31.37	30439.06	137405.57	167844.63
Shuttle 2	665.46	24.05	1324.05	36.61	1989.51	55.01	68877.25	611632.73	680509.98
Shuttle 3	511.82	13.67	1209.21	41.51	1721.04	45.95	21742.19	803539.26	825281.45
Satellite 1	998.66	28.23	353.04	9.30	1351.70	35.61	206935.63	38895.15	245830.78
Satellite 2	779.43	28.03	1483.74	40.80	2263.17	62.23	196206.46	160278.60	356485.05
Satellite 3	602.80	16.03	1368.90	46.67	1971.70	52.42	151001.36	176695.36	327696.72

Figura 58 Tabella con statistiche di simulazione

3.2 Dati relativi all'energia.

Come si è potuto notare, in tutte le tabelle in cui si raccolgono dati, sono presenti alcune voci relative all'energia, poiché è stata impostata un'analisi energetica, sfruttando le dinamiche di simulazione ed i vantaggi dell'ambiente Flexsim, appoggiandosi su alcuni studi teorici.
Il modello analitico prevede come input le caratteristiche cinematiche dei veicoli, cioè accelerazione e velocità massima raggiungibili dell'ascensore, degli shuttle e dei satelliti e la loro massa. È facilmente deducibile come questi valori, agevolmente modificabili dall'utente attraverso la GUI del modello, influenzino i risultati energetici trovati e come l'analisi effettuata possa essere valida solo per le caratteristiche scelte nelle simulazioni lanciate. Tuttavia, lo scopo principale è quello di testare un nuovo modo di analizzare tali dati, sfruttando un software nato per simulazioni ad eventi discreti e non per analisi pura.
L'energia consumata dai veicoli tiene conto di un moto uniformemente accelerato e tutti i movimenti iniziano con il veicolo fermo. Dunque, è calcolabile il tempo impiegato da un veicolo, con accelerazione costante, per raggiungere la velocità massima, cioè quella di regime che sarà costante prima della decelerazione.

$$T = \frac{v}{a} \ [s]$$

In cui:

- T è il tempo calcolato in secondi [s];
- v è la velocità [$\frac{m}{s}$];
- a è l'accelerazione [$\frac{m}{s^2}$].

Logicamente tale calcolo, essendo differenti le caratteristiche dei vari veicoli, è stato ripetuto per le tre categorie (ascensore, shuttle e satellite) ed i valori sono stati riferiti all'asse in cui tale veicolo lavora e salvati come variabili globali, in quanto utilizzano caratteristiche assolute che non si modificheranno durante la simulazione. Dunque, tali variabili sono:

- T_x per gli shuttle;
- T_y per i satelliti;
- T_z per l'ascensore.

Tali tempi calcolati sono le discriminanti per designare le dinamiche di moto di ogni viaggio effettuato dai veicoli, i quali sicuramente hanno una fase iniziale di accelerazione e una finale di decelerazione. Poiché l'accelerazione e la decelerazione sono modellati con lo stesso valore, tali fasi avranno la stessa durata, pari proprio a T e sono calcolabili le potenze impiegate su unità di massa $[\frac{W}{Kg}]$:

- Per la fase di accelerazione $P_A = (a \cdot f_r + g \cdot c_r) \cdot \frac{v_{max}}{\eta}$;
- Per la fase di decelerazione $P_B = (a \cdot f_r - g \cdot c_r) \cdot \frac{v_{max}}{\eta}$;

L'accelerazione è moltiplicata per un fattore di resistenza di massa f_r ed è un fattor comune alle due fasi. La contrapposizione tra accelerazione e frenata è sottolineata nel secondo fattore, rappresentato dall'accelerazione di gravita moltiplicata per il coefficiente di frizione c_r. La velocità massima è la velocità raggiunta dal veicolo prima di iniziare a rallentare e può essere al massimo pari alla velocità a regime del veicolo stesso, calcolata come rapporto tra l'accelerazione ed il tempo di percorrenza. È presente un ultimo coefficiente di correzione, rappresentante l'efficienza del sistema di trasmissione η. Oltre queste due fasi, è possibile la presenza di una terza fase intermedia, quando il veicolo raggiunge la sua velocità nominale di regine e inizia un percorso a velocità costante, con un contributo:

$$P_C = g \cdot c_r \cdot \frac{v}{\eta}$$

Nella fisica cinematica esiste una relazione diretta tra spazio e tempo, con leggi precise di calcolo, per cui è ci sono due verifiche possibili per capire se il veicolo in azione è riuscito nella fase di accelerazione a raggiungere la velocità nominale prima di decelerare: banalmente una valuta il tempo e l'altra le distanze percorse.
Nell'ambiente di simulazione Flexsim, risulta più intuitiva la verifica sul tempo, in quanto è più agevole cronometrare le attività che calcolare lo spazio percorso. Se il tempo totale, indicato con T_t

di un viaggio è superiore a due volte il tempo caratteristico del moto uniformemente accelerato *T,* il veicolo avrà una porzione di moto uniforme a velocità costante. In particolare, l'energia consumata in ogni processo è calcolabile moltiplicando le potenze trovare per i rispettivi tempi $[\frac{J}{kg}]$:

$$E = \begin{cases} P_A \cdot \frac{T_t}{2} + P_B \cdot \frac{T_t}{2} & \boldsymbol{se}\ T_t < 2T \\ P_A \cdot \frac{T_t}{2} + P_c \cdot (T_t - 2T) + P_B \cdot \frac{T_t}{2} & \boldsymbol{se}\ T_t > 2T \end{cases}$$

Logicamente i calcoli sono completati moltiplicandoli per le masse dei veicoli, considerando che l'ascensore nei suoi movimenti trasporta anche uno shuttle ed un satellite, lo shuttle nel movimento ha sempre a bordo il satellite e, quando necessario, si deve tener conto anche nel peso dell'item.
Unica eccezione a queste leggi di calcolo è rappresentata dal movimento in discesa dell'ascensore, in accordo con l'attrazione gravitazionale e dunque ai fini di questa analisi trascurabile.

4. ANALISI DATI

Per poter affrontare qualunque discussione ed analisi è necessario un modello di riferimento. In particolare, in questo lavoro, si è studiata la situazione di un'azienda italiana, impegnata nell'area *food*. Al momento il suo sistema logistico è basato su un magazzino composta da due rack, con sei livelli, di cui due completamente interrati, e 38 canali per ogni scaffalatura. La scaffalatura di destra ha una capacità di 12 item e quella di sinistra di 14, ma sono presenti alcuni impedimenti strutturali, che riducono la capacità di alcune baie. Ad oggi, il processo è alimentato con un sistema AVS/RS composto da un ascensore ed una singola coppia shuttle/satellite e per le analisi sono state utilizzate le caratteristiche di questi veicoli:

- Accelerazione ascensore: 0.35 m/s^2;
- Velocità massima ascensore: 14 m/min;
- Accelerazione shuttle: 0.55 m/s^2;
- Shuttle maximum speed: 120 m/min;
- Accelerazione satellite: 0.60 m/s^2;
- Velocità massima shuttle: 72 m/min.

4.1 Analisi Preliminare

Il primo obiettivo concreto è stato quello di testare il comportamento ed il funzionamento delle logiche decisionali implementate nel software e la risposta del modello alla variabilità dei dati in ingresso. È stata condotta una campagna di 400 simulazioni, in cui sono stati simulati 4 giorni di lavoro continuo ed in modo casuale si sono create differenti combinazioni di input, con lo scopo di designare per ognuna un possibile modo di funzionare di un reale magazzino.
I parametri variabili sono molti, in primo luogo i criteri di stoccaggio e di retrieval ed il numero di veicoli utilizzati. Un'attenzione particolare è rivolta al *Fill Rate* iniziale. È la percentuale di capacità piena all'istante zero della simulazione. Si riesce così a verificare le differenze di prestazioni tra un magazzino inizialmente quasi vuoto ed uno quasi completamente pieno. Il primo feedback del simulatore è stato il tempo computazionale di calcolo, in quanto nelle fasce 0 - 10 % e 90 – 100% questo aumenta notevolmente, non restituendo risultati diversi da giustificare un'analisi specifica. Per questo motivo si è deciso di limitare l'analisi alla fascia 10 – 90%, estendendo per similitudine il comportamento alle percentuali non coperte. Altri parametri variabili sono il tempo medio di richiesta delle operazioni. Nello specifico la variabile *Storage Rate Mean* indica mediamente, con un valore di scostamento *Storage Rate Variance,* ogni quanti secondi un item è creato e richiede un'operazione di stoccaggio. Tale condizione graficamente è visibile nella baia di stoccaggio, in cui si depositano gli item in attesa di essere prelevati dai veicoli e depositati. Alla fine della simulazione, è plausibile che un certo numero di item sia ancora in coda e non sia stato allocato nelle scaffalature. Il primo KPI per analizzare il comportamento è rappresentato proprio dalla percentuale di storage effettuati, calcolata come rapporto tra il numero di item stoccati e le richieste totali. Ragionamento analogo è applicabile per le operazioni di retrieval, in cui ogni arco di tempo, identificato dal *Retrieval Rate Mean* e la sua variazione *Retrieval Rate Variance*, arriva una richiesta di ripresa dal magazzino, identificato da un token del Process Flow. Logicamente, anche i token devono aspettare la disponibilità dei veicoli e restano in coda nelle attività del processo. Anche in questo caso è identificabile un KPI, calcolato come rapporto tra le operazioni realmente eseguite e quelle richieste dal sistema. In particolare, nelle simulazioni, in accordo con i dati raccolti dal caso studio, lo *Storage Rate Mean* e il *Retrieval Rate Mean* variano tra 100 e 250 secondi. Invece per le variazioni abbiamo un valore compreso tra 0 e 100 per gli storage ed un valore tra 0 ed il valore di media assegnato per i retrieval, in quanto la richiesta di ripresa è molto più variabile rispetto a quella degli stoccaggi, dipendente dal flusso di produzione dell'azienda. Dalle 400 simulazioni, si è estratto un tempo ciclo medio ed un consumo medio per ogni item, sia per lo storage che per il retrieval.

Tempo Ciclo Medio Storage [s]	60,64
Tempo Ciclo Medio Retrieval [s]	78,90
Consumo medio Storage [Wh]	86,73
Consumo medio Retrieval [Wh]	27,80

Figura 59 Tabella valori medi delle simulazioni

Nei grafici seguenti si è messo in evidenza il comportamento delle percentuali di operazioni effettuate rispetto al valor medio atteso per l'arrivo delle richieste, con in ascissa il valor medio ed in ordinata i KPI calcolati.

Sui grafici seguenti sono identificate tutte le simulazioni svolte con dei punti neri ed è un buon test per verificare la corretta logica di funzionamento.

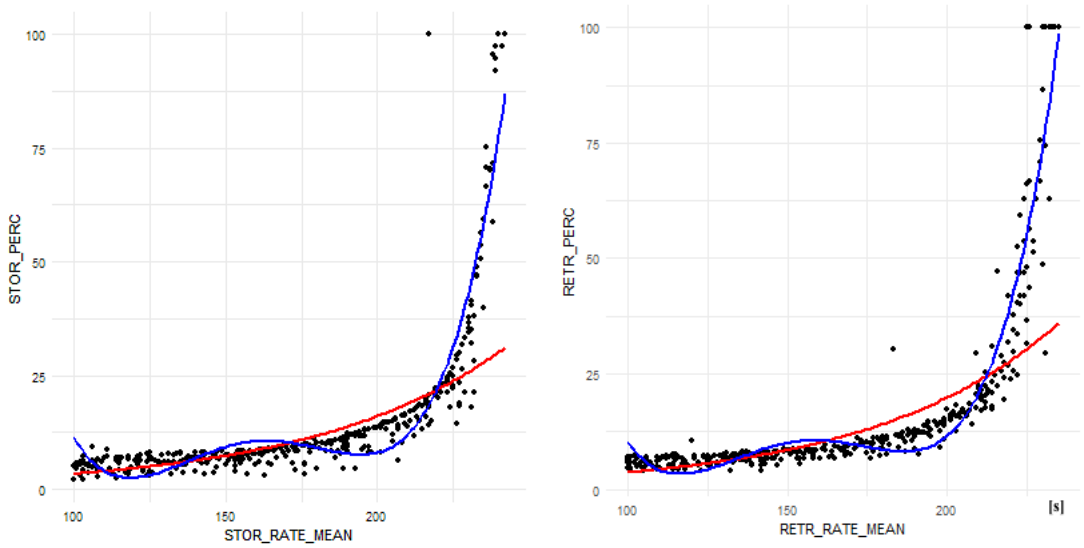

Figura 60 Grafici KPI al variare del tempo medio delle richieste

Infatti, al crescere della media di arrivo delle richieste, è attendibile che il sistema abbia una gestione migliore, in quanto ha più tempo per completare un'operazione prima che arrivi una nuova richiesta e la coda aumenti. Si potrebbe pensare che, per le simulazioni in cui il tempo medio di richiesta è maggiore del tempo ciclo delle operazioni, si riscontri sempre un'efficienza del 100 %, affermazione non vera poiché i due rami del Process Flow e di conseguenza i processi di Storage e Retrieval, sono gestiti contemporaneamente ed utilizzano gli stessi veicoli. Tranne pochi punti di discontinuità, all'aumentare della media, le percentuali di storage e retrieval aumentano con delle leggi. La curva rossa è una regressione esponenziale che, però, in nessuno dei due processi riesce a rappresentare l'andamento trovato. La curva blu, invece, è una polinomiale di quarto grado ed è una buona approssimazione della distribuzione dei risultati e restituisce due leggi precise:

- *Storage*, $y = 15.3 + 237.22 \cdot x + 181.15 \cdot x^2 + 132.24 \cdot x^3 + 97.42 \cdot x^4$;
- *Retrieval*, $y = 17.3 + 279.80 \cdot x + 210.00 \cdot x^2 + 148.12 \cdot x^3 + 96.41 \cdot x^4$.

È stata effettuate un'ulteriore valutazione di tipo generale, con in ascissa lo *Storage Rate Mean* ed in ordinata il *Retreival Rate Mean*: la grandezza dei punti indica la percentuale di stoccaggi effettuati ed il variare del colore la percentuale di riprese dal magazzino asservite.

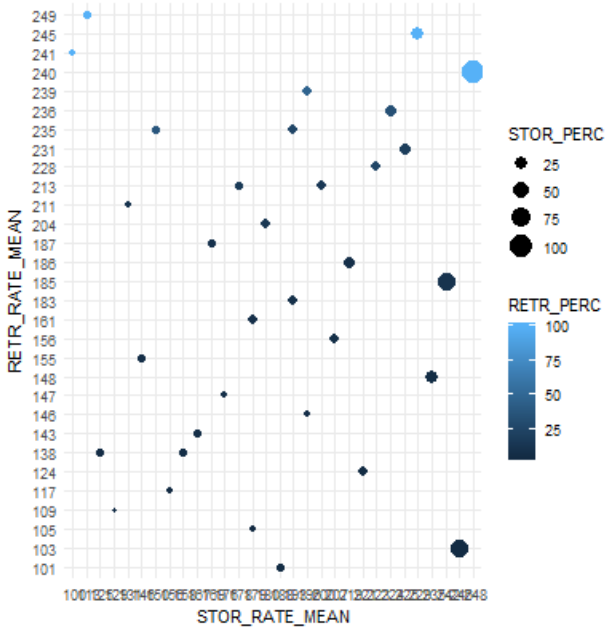

Figura 61 Grafico KPI con variazione contemporanea di Storage/Retrieval Rate Mean

Come atteso dai risultati dei grafici precedenti, le simulazioni migliori si collocano nella zona in alto a destra, in cui le due medie hanno valori più alti.

4.2 Valutazioni criteri di stoccaggio

In ordine di compiere analisi più dettagliate, è necessario determinare i criteri migliori da utilizzare per le simulazioni. Per il retrieval la scelta è stata puramente teorica, tenendo conto anche dell'azienda del caso studio. Infatti, soprattutto nell'ambito alimentare, i prodotti stoccati hanno una scadenza e, comunque, sono soggetti a deterioramento nel tempo. Di conseguenza, il criterio più adatto è quello denominato *for Creation*, in cui si svuota il canale con l'item più vecchio, cioè creato prima. Identificato il criterio per la ripresa, si è studiato quale fosse il criterio più performante per gli

stoccaggi. Dalla campagna di 400 simulazioni, si sono considerate solo quelle in cui effettivamente si fosse usato il criterio prescelto per il retrieval, ottenendo 99 simulazioni effettive. Questo dato certifica anche la buona distribuzione dei dati in input, essendo circa un quarto delle simulazioni effettuate. I KPI prescelti sono gli stessi dell'analisi precedente, in quanto restituiscono una descrizione puntuale della risposta del sistema. In ogni grafico sono state plottati i risultati delle varie simulazioni, identificate con un marker di differente colore, in base al criterio di stoccaggio utilizzato. Nel grafico seguente in ascissa ed ordinata ci sono i due KPI prescelti, cioè la percentuale di storage e retrieval completati. È evidente come la zona con le prestazioni migliori sia in alto a destra, in cui entrambi i KPI hanno un valore elevato, prossimo al 100%, dove c'è una prevalenza di colore blu, assegnato al criterio 1, cioè al *Closest Channel.*

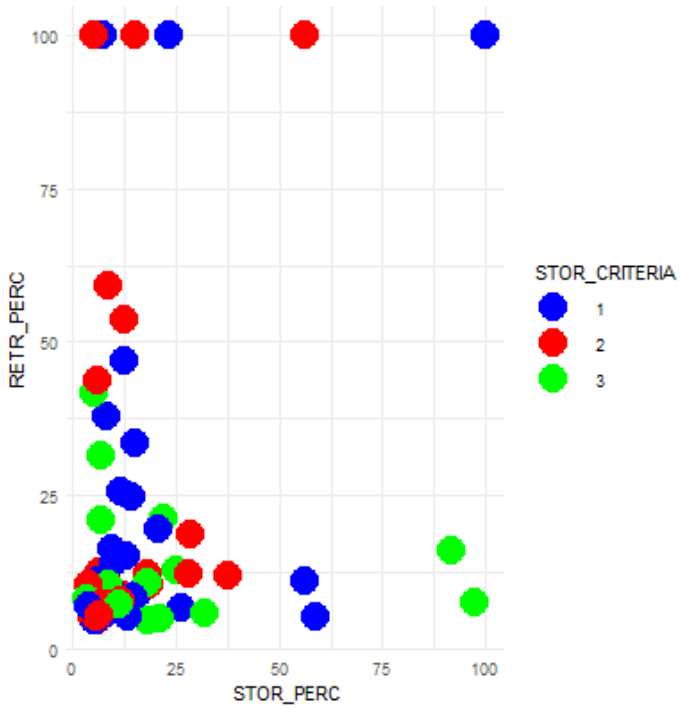

Figura 62 Grafico criteri e KPI

A conferma della bontà di questo criterio, è stato valutato anche l'impatto energetico dei criteri, valutando l'energia totale spesa in ogni simulazione e la solita percentuale di operazioni completate.

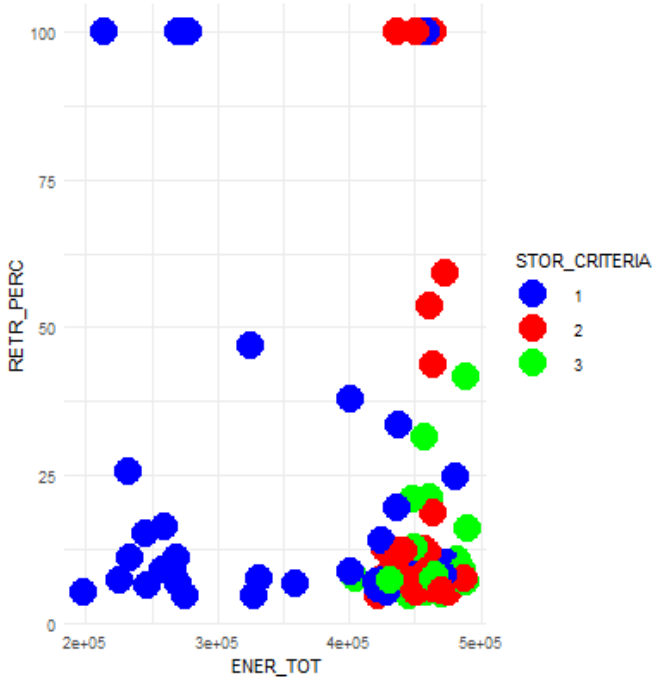

Figura 63 Grafico con valutazione energia totale

Qui i risultati sono molto chiari, il criterio *Closest Channel* ha un impatto di consumo molto basso rispetto agli altri criteri, deduzione in linea con le aspettative stesse del criterio. È da sottolineare come i risultati energetici del criterio *Closest Floor* siano molto alti. La differenza sostanziale tra i due criteri, trascurando tutte le variabilità possibili, è nel movimento tra i livelli dell'ascensore, che con il suo grosso peso impatta molto nei consumi.

Nei seguenti grafici si analizza come variano i KPI al variare del fill rate iniziale. Nella risposta del modello simulato non è possibile riscontrare nessun comportamento particolare, dimostrazione con esiste una correlazione tra la condizione iniziale e la percentuale di richieste di deposito o prelievo soddisfatte, prova aggiuntiva della corretta gestione delle attività da parte del Process Flow.

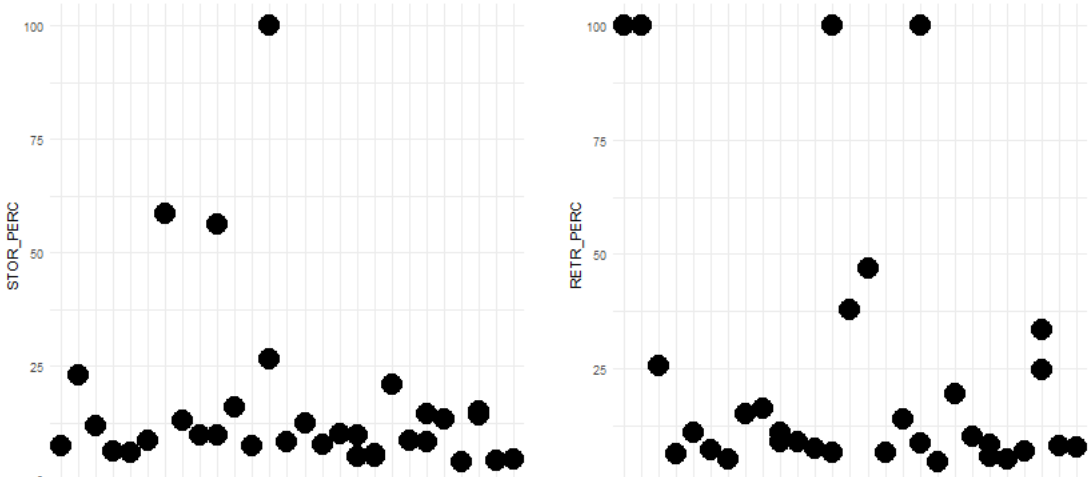

Figura 64 Grafici KPI al variare del fill rate iniziale

4.3 Analisi sul caso studio

Determinati i due criteri, il *Closest Channel* per lo storage e il *For Creation* per il retrieval, si è analizzata la situazione attuale del magazzino del caso studio, con l'obiettivo di capire dove sarebbe più opportuno investire per migliorare le sue prestazioni. In particolare, si è studiato una soluzione con fill rate iniziale del 75%, con uno o due veicoli, al variare delle loro velocità. Gli scenari risultanti da tali scelte sono 18, in cui le velocità dello shuttle e dell'ascensore sono impostate al 75%, al 100% ed al 125% del valore reale. Per ogni scenario sono state fatte 3 ripetizioni, considerando il risultato medio delle simulazioni [18].

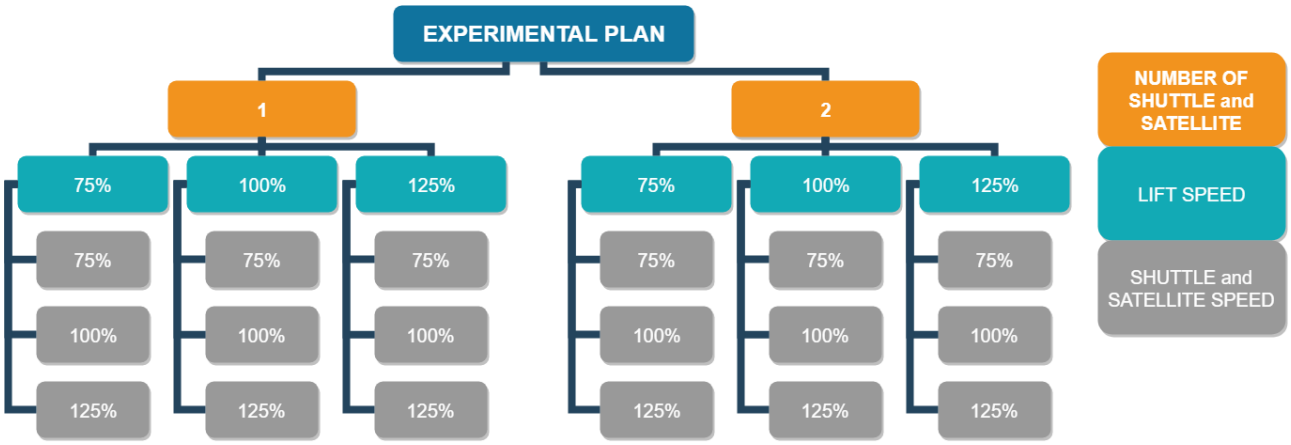

Figura 65 Experimental Plan

In prima analisi è necessario subito identificare se sia migliore la configurazione con una sola coppia shuttle/satellite o due. Per fare ciò si sono considerati 3 KPI, cioè le percentuali di storage e retrieval effettuati e il consumo totale di energia. Nel primo grafico, i risultati sono abbastanza ovvi, poiché la parallelizzazione delle attività comporta un maggior numero di richieste completate con due coppie di veicoli rispetto ad uno. Nello specifico, con due shuttle/satellite si riesce a realizzare un'efficienza sempre superiore al 95% per le riprese dal magazzino e superiore all'87% per i depositi. Invece, con una coppia di veicoli è impossibile avere performance superiori all'85% dei retrieval, con risultati migliori per gli storage. La differenza esiste, ma non è però così marcata. Il secondo grafico fornisce, invece, considerazioni più significative. Si nota come il consumo energetico non dipenda dal numero di operazioni effettuate, ma nel limitare il numero di movimenti. Il grafico è divisibile in due comportamenti ben definiti. Con una coppia di veicoli, si hanno consumi crescenti con la percentuale di retrieval effettuati. Tali valori sono specchiati quasi simmetricamente sulle configurazioni triangolari, con due coppie shuttle/satelliti, in cui però la percentuale di retrieval è pari o superiore al 95%. Si deduce che, a parità di energia consumata, con due coppie di veicoli si riesce ad avere

un'efficienza superiore del 10% almeno, segno che l'utilizzo parallelo dei veicoli è ben strutturato e bilanciato. Focalizzando l'attenzione sulle configurazioni triangolari, è facile identificare le soluzioni migliori in alto a sinistra, dove con un valore di energia limitato si riesce a completare la maggior parte delle richieste arrivate al sistema.

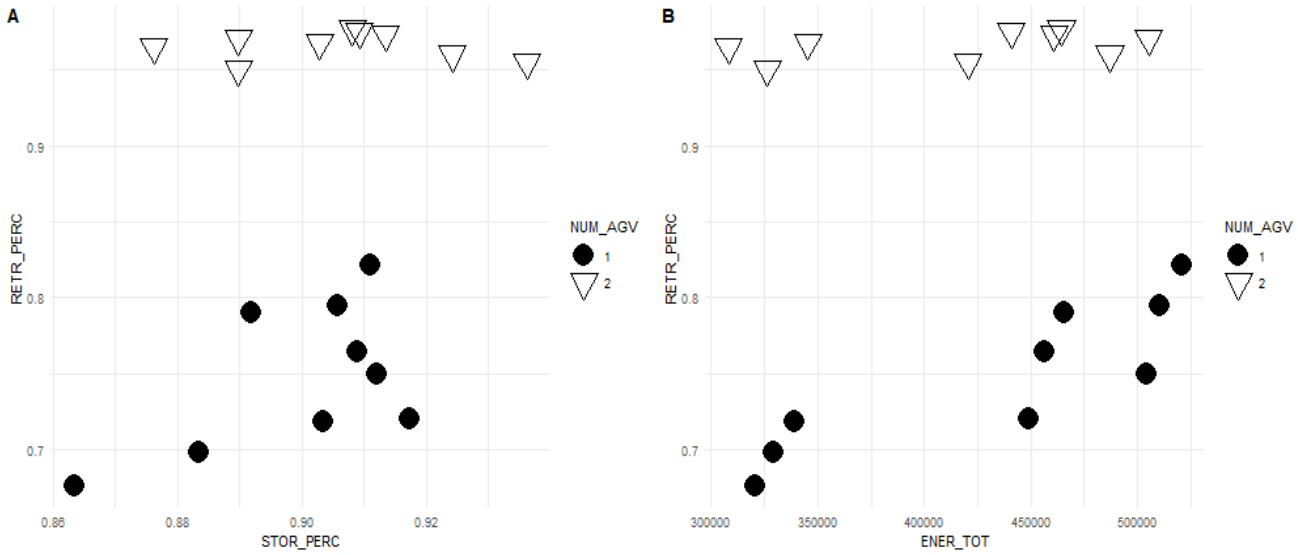

Figura 66 Grafici con differenze tra singola o doppia coppia di veicoli

Nei seguenti grafici si analizzano i comportamenti del sistema al variare delle velocità massima dello shuttle e dell'ascensore, con un grafico in cui sono incluse insieme informazioni riguardo alle percentuali di richieste soddisfatte e l'energia totale. I risultati suggeriscono che all'aumentare della velocità degli shuttle aumentano i due KPI principali, con un consumo di energia che non varia significativamente. Invece, il grafico relativo alla velocità dell'ascensore mostra una risposta del sistema simile per i tre valori di velocità. Tale poca variabilità induce a pensare che la scelta della sequenza di prenotazione dell'ascensore e le sue logiche decisionali siano ben impostate, permettendogli di distribuire correttamente le sue attività per soddisfare il maggior numero di richieste.

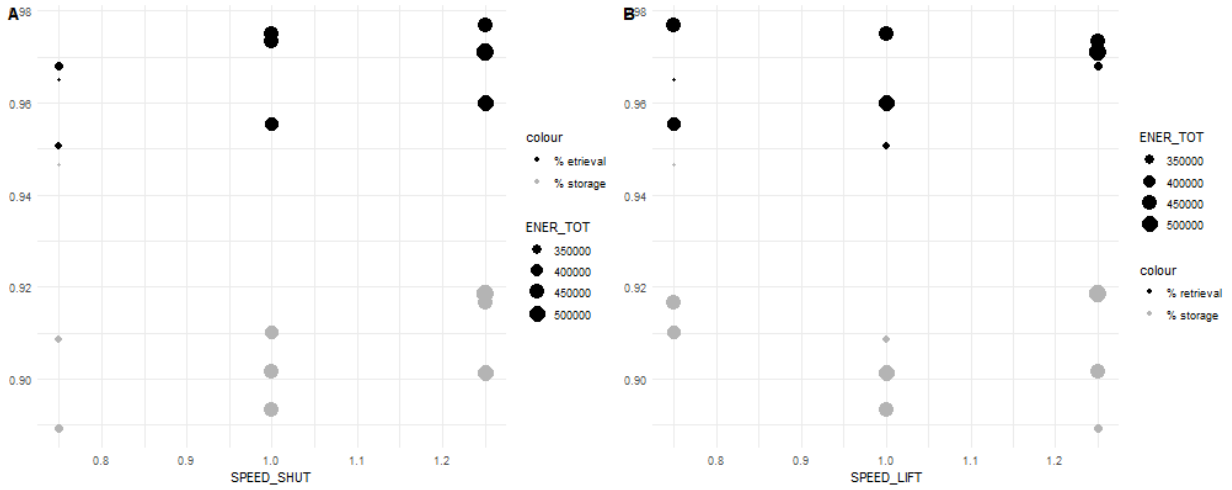

Figura 67 Grafici valutazione velocità shuttle ed ascensore

Per la scelta della velocità dell'ascensore si è impostata, allora, una più definita valutazione dell'energia consumata, media del consumo delle simulazioni con una determinata velocità. Dal grafico in figura, si nota come il comportamento non sia stabile o lineare, con tuttavia un consumo molto ridotto per la velocità più bassa. Quindi, in virtù di questa analisi, una buona soluzione prevede shuttle con velocità massima ed ascensore con velocità minima, per garantire una riduzione di energia consumata.

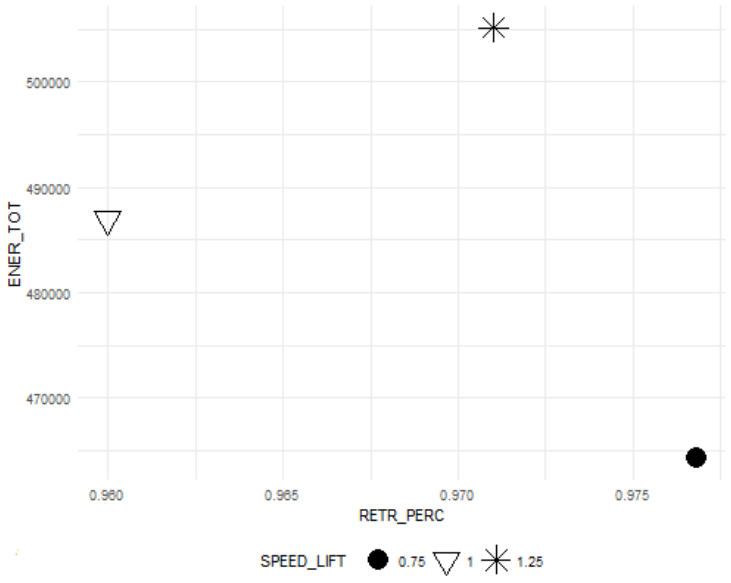

Figura 68 Grafico valutazione energia ascensore

Nel nostro percorso di analisi, si è giunti alla scelta di un determinato scenario. Nell'ultimo grafico tridimensionale, si verifica la scelta fatta valutando la percentuale di utilizzo dei veicoli, calcolata come rapporto tra tempo di attività e tempo totale di simulazione.

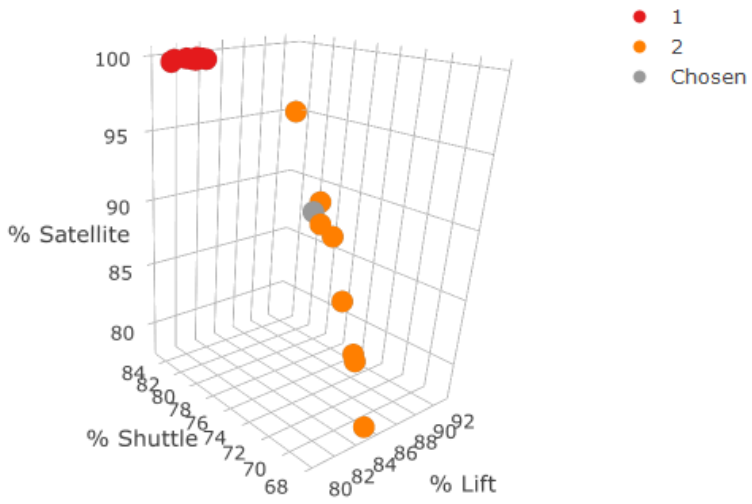

Figura 69 Grafico percentuali di utilizzo veicoli

Il tempo di attività è calcolato dall'istante in cui il veicolo è prenotato per un'operazione di storage o retrieval, fino a quando termina tale processo. Con una coppia di veicoli il satellite è in attività per il 100% del tempo, comportamento immaginabile visto l'alto numero di richieste da soddisfare. Anche in questa analisi, si sottolinea come la doppia coppia di veicoli sia più performante, distribuendo in modo più omogeneo il lavoro sui diversi veicoli ed aumentano l'efficienza dello stesso ascensore, sintomo di una maggiore distribuzione del lavoro e delle percentuali di richieste soddisfatte. La soluzione scelta mostra un ottimo comportamento e può essere validata, alla luce di tutte le considerazioni fatte.

Conclusioni

Nell'ambito produttivo, la simulazione ad eventi discreti sta assumendo un'importanza sempre maggiore, sia come strumento di analisi che come strumento di vendita. La logistica, in particolare, si sposa perfettamente con i principi cardine di tale strumento. Il software Flexsim permette di esplorare tutte le soluzioni esistenti sul mercato, grazie ad una libreria di oggetti molto fornita e la possibilità di effettuare una forte customizzazione, sia grafica che logica.
Nel modello costruito si possono identificare due strutture fisiche fondamentali:

- I *rack*, la struttura fissa in cui depositare e ritirare gli item. Si sviluppa su tre dimensioni, identificate dai livelli, dai canali e dalla loro profondità.
- Il sistema *AVS/RS,* sistema di trasporto automatico composto da shuttle, satelliti ed ascensore.

Su queste strutture, il modello ha raggiunto un buon livello di flessibilità, permettendo la costruzione di magazzini con differenti configurazioni strutturali. Anche la forma stessa può, in modo artificioso, essere personalizzata, inserendo con una matrice il numero di posizioni accessibili in ogni determinato canale, simulando magazzini con scaffalature asimmetriche. Per quanto riguarda i veicoli, la possibilità di cambiare i parametri come velocità ed accelerazione, consente la valutazione di differenti settaggi, valutandone le varie prestazioni.
Le logiche decisionali presenti nel modello tentano di dare una risposta alle diverse esigenze del mondo produttivo, permettendo la scelta tra differenti criteri di stoccaggio e ripresa dal magazzino. Nonostante ciò, esistono sicuramente altri criteri da poter inserire: i codici script implementati nelle varie decisioni, hanno una struttura molto solida da permettere, in futuro, con poche modifiche l'impostazione di nuovi criteri, sia per lo storage che per il retrieval.
Oltre gli aspetti di costruzione del modello e di simulazione, nel Process Flow di Flexsim si è installata una buona piattaforma per l'analisi dei dati. Alla fine di ogni simulazione lanciata, è possibile estrarre in modo semplice ed intuitivo delle tabelle in cui sono indicati molti parametri, in termini di tempo, energia e percentuale di utilizzo dei veicoli, che si prestano per varie e molteplici analisi. In questo modo, il simulatore può essere un ottimo strumento per decisioni, sia prima di un investimento importante, sia per correggere alcuni parametri in impianti già esistenti.
Altro punto di forza dell'infrastruttura del modello è l'usabilità da parte di utenti poco pratici o che non conoscono l'ambiente di simulazione Flexsim. L'interfaccia utente inserita permette l'impostazione di tutti i parametri discussi, guidando l'utente dalla costruzione fisica del modello, alla scelta dei criteri e del riempimento inziale, fino al lancio vero e proprio della simulazione.

Bibliografia

[1] Tecnologico, Logistica integrata Riduzione costi e aumento del valore Corrado Cerruti Università di Roma Tor Vergata.

[2] C. J. Malmborg, Conceptualizing tools for autonomous vehicle storage and retrieval systems," Int. J. Prod. Res., vol. 40, no. 8, pp. 1807–1822, 2002..

[3] P. Kuo, A. Kristnamurthy e C. Malmborg, "Design models for unit load storage and retrieval systems using autonomous vehicle technology and resource conserving storage and dwell point policies," Appl. Math. Model., vol. 31, no. 10, pp. 2332–2346, 2007..

[4] B. Y. Ekren, S. Heragu, A. Kristhnamurthy e C. J. Malmborg, "Simulation based experimental design to identify factors affecting performance of AVS/RS," Comput. Ind. Eng., vol. 58, no. 1, pp. 175–185, 2010..

[5] S. Heragu e B. Ekren, "Performance comparison of two material handling systems: AVS/RS and CBAS/RS," Int. J. Prod. Res., vol. 50, no. 15, pp. 4061–4074, 2012..

[6] B. Ekren e S. Heragu, "Simulation based performance analysis of an autonomous vehicle storage and retrieval system," Simul. Model. Pract. Theory, vol. 19, no. 7, pp. 1640–1650, 2011..

[7] S. Heragu, X. Cai, A. Krishnamurthy e C. Malmorg, Analytical models for analysis of automated warehouse material handling systems," Int. J. Prod. Res., vol. 49, no. 22, pp. 6833–6861, 2011..

[8] B. Zou, X. Xu, Y. Gong e R. De Koster, "Modeling parallel movement of lifts and vehicles in tier-captive vehicle-based warehousing systems," Eur. J. Oper. Res., vol. 254, no. 1, pp. 51–67, 2016..

[9] G. D'Antonio, M. De Maddis, J. Bedolla, P. Chiabert e F. Lombardi, "Analytical models for the evaluation of deep-lane autonomous vehicle storage and retrieval system performance," Int. J. Adv. Manuf. Technol., vol. 94, no. 5–8, pp. 1811–1824, 2018..

[10] T. Lerher, "Design of Experiments for Identifying the Throughput Performance of Shuttle-Based Storage and Retrieval Systems," Procedia Eng., vol. 187, pp. 324–334, 2017..

[11] M. Glatta, G. Kasakow e J. Auricha, Combining physical simulation and discrete-event material flow simulation.

[12] C. Harrel e K. Tumay, Simulation Made Easy, Engineering & Management Press.

[13] J. W. a. R. T. Schmidt, J. Schmidt e R. Taylor, Simulation and Analysis of Industrial Systems, Richard D.Irwin, Homewood, 1970.

[14] M. Averill Law, Simulation Modelling & Analysis, Mc Graw Hill, 2007.

[15] Flexcon s.r.l. Strada del Drosso, 33/8 I–10135 TORINO - Italy www.flexcon.it, Basic Training Course FlexSim.

[16] M. W. Barnett, "In Business Process Management", Model. Simul. Bus. Process Manag., pp. 1–10, 2003..

[17] Synthesizer Direct Digital Connect, «FlexSim User Manual,» Surgicase, 2010.

[18] G. Bruno, E. Traini, P. Chiabert e F. Lombardi, «Configuration of a production-integrated AVS/RS through discrete event simulation» *XIV Convegno dell'Associazione Italiana Tecnologie Manifatturiere - AITEM,* vol. (in press), 2019.

Indice delle figure