POLITECNICO DI TORINO

Collegio di Ingegneria Gestionale

Corso di Laurea Magistrale in Engineering and Management
Percorso Production

Master's degree thesis

Design of Innovative Transport Systems for automated warehouses

Supervisors:

Prof. Franco Lombardi
Prof. Giulia Bruno
Prof. Alberto Faveto

Candidate:

Sonia Ariano

Academic Year 2020/2021

Abstract

Warehouses' performance evaluation has often been a relevant topic of research in the field of Logistics, moreover with the increasing importance acquired by digitalization and robotics, the research has found different ways to broaden itself. In an era where e-commerce is expanding quickly, customers demand the right product at the right time and behind the ability of companies to fulfil customer orders, there is an efficient way of managing incoming orders, picking, retrieval and storage activities. Nowadays automatic warehouses, which have to perform such activities, are widely used and they allow, in most cases, to increase the performance of the warehouse by improving its order throughput and reducing orders cycle time. One of the most efficient automatic systems, aimed at fulfilling orders, is the automatic vehicle storage and retrieval system, also called AVS/RS. The present study wants to evaluate the performance of such a system with an additional feature designed by the Italian company Eurofork: a robotic arm mounted on the shuttle. In this way, in addition to retrieval and storage activities, the system will also be able to perform picking activities. The system is evaluated through discrete event simulation and tested under different scenarios and under different storage policies to see what are the best environment, best storage policy, and best external settings that maximize its performance.

Keywords – AVS/RS, Discrete Event Simulation, Storage Policies, Innovative automated order picking system, Order Picking activity

Acknowledgement

I would like to express my special thanks of gratitude to the professors Franco Lombardi and Giulia Bruno from the Department of Management and Production Engineering (DIGEP) at Politecnico di Torino for their constant support throughout the thesis.

I would also like to acknowledge the valuable time and precious guidance offered by Alberto Faveto during the practical implementation of the study.

Special thanks go to my family and my loved ones for their constant encouragement and support during these years, I will always be grateful for your unconditional love.

Table of Contents

1.Introduction

Warehouses are considered crucial elements of the supply chain network, as they are the physical space where goods are stored to guarantee a fluid and uninterrupted flow of products along the supply chain (Shashank Kumar et al., 2021). The first studies related to warehouses are mainly focused on the identification of the optimal position for the facility that could lead to cost reduction, however, as the interest in warehouses increased and as the market changed, the focus of studies moved towards the concept of automated warehouse through the use of different innovative material handling systems (Shashank Kumar et al., 2021). Order picking is considered one of the most important activities happening inside warehouses and it is estimated that it accounts for around 55% of all operating costs as well as 70% of the overall time dedicated to warehouse activities (Josip Habazin et al., 2017). The continuous growth of e-commerce and consequently the increasing need to meet the customers' demands in the most accurate way in terms of content and time, is putting the warehouse system under a lot of pressure (Shashank Kumar et al., 2021). Finding new automatized ways to handle material could lead to consistent improvements such as shorter delivery times, higher accuracy, better space utilization, lower costs, which are elements that increase the service offered to the customer and improve the company's competitiveness (Gino Marchet et al., 2015). Because of the aforementioned reasons, research and studies are actively focusing on the evaluation of new technology which could lead to an overall improvement of the order picking activities (Yasmeen Jaghbeer et al., 2020).

1.1 Objective of the study

This study aims at evaluating the performance of an innovative system that could further advance the activities of autonomous storage, retrieval, and picking. The new automated system has been proposed by the company Eurofork, an Italian company that is continuously gathereing know-hows and competencies in the context of automated material handling systems. The new automated system will consist of a shuttle with a robotic arm installed on it and it will therefore try to combine the activities already performed by an AVS/RS with the activities of picking to create personalized pallets in the

most modular and flexible way (Figure 1). The technology will be based on the already commercialized ESMARTSHUTTLE®, which is able to perform activities similar to an AVS/RS system, however, a robotic arm is installed on the shuttle. This allows to enhance the potentiality of the system even more by enabling it to create mixed pallets with different units, based on the demand orders. This structure will equip the system with the skill to handle stock units of different dimensions, from pallets to small items. Moreover, the successful implementation of this system will eliminate the need to have a dedicated picking area where operators manually pick the relevant units to satisfy the orders. With the integration of different order picking systems, handling of pallets, and handling of small totes, the new technology aims at introducing a set of revolutionary advantages in the market compared to the solutions currently used:

- Increased efficiency
- Lower energy consumption
- Lower resource allocation costs and maintenance cost
- The entire order picking activity will be performed in a single plant
- Higher flexibility

The performance of this new system will be evaluated using a simulation model in FlexSim which simulates a situation in which the new innovative technology is used to perform the picking activity. The study will try to identify the best storage policy to use in combination with the new picking technology as well as the best design in terms of rack configuration so that a better overall performance can be achieved. The study will therefore try to run several experiments to identify the best rack configuration as well as the best storage policy.

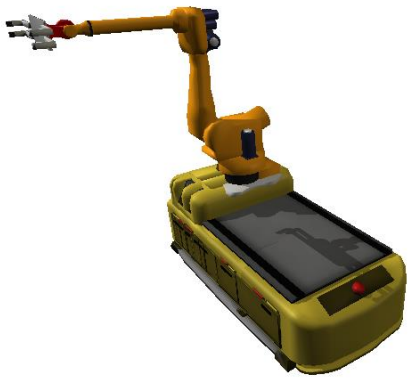


Figure 1: Vehicle with robotic arm mounted on the shuttle

1.2 Organization of the thesis

The study will start with an extensive literature review which will focus on describing the main and most used automated systems for storage, retrieval and picking. The literature review will then show how Discrete Event Simulation (DES) has been used in the past in the context of automated storage, retrieval and picking systems with specific focus on AVS/RS.

The following section will focus on the description of the main functionalities of the simulation software FlexSim that has been extensively used for this study.

The logic behind the different storage policies that will be part of the study will then be taken into consideration. The different storage policies, namely the Random, Class-Based Storage, Dedicated Slot storage, Storage by weight and storage by Association rules will be described and a graphical flow chart will show the modelling logic behind the activities of picking in an automated warehouse.

After that, the study will describe the model that has been developed in FlexSim with focus on its implementation and the output expected from it. The paper will then show the results coming from the experiments run in the FlexSim model which simulates arrivals of products and arrivals of demands for specific combination of products stored in stock units. Through simulation it is possible to evaluate how the performance changes as different rack configurations and different storage strategies are employed. The analysis of the performance will give insights on the potential benefits that this new system can bring into the warehouse organization.

The study will then conclude with the results of the exercises, and it will provide some limitations of the present study and further developments for a future research.

2. Literature review

The present literature review will first introduce a comprehensive presentation of the most important automated systems in warehouses, and it will later focus on studies that showed

interest in evaluating the performance of automates systems (mainly AVS/RS) through Discrete Event Simulation.

2.1 Automated warehouse systems for storing, retrieving, and picking activities (OPS -Order Picking Systems)

Automated Storage and Retrieval Systems, also abbreviated with AS/RS, have been introduced for the first time in the 1950s and right from the beginning they played a crucial role in ensuring substantial improvements in inventory management and material handling issues, increasing companies' flexibility and competitiveness (Farah Hanani M.K. et al., 2016). Among the many advantages that come with AS/RS it is important to underline the increasing in accuracy and material handling control, a better utilization of inventory space, equipment and better responsiveness and speed in storing and retrieving unit loads (Vasili M.R. et al., 2012). The automated solutions introduced over the years allowed to automatize activities such as storing, retrieving, and picking which are the focus of this paper. Storing activities refer to the need to put away SKUs (Stock Keeping Units) in specific locations inside a warehouse; the stocking activities can follow different policies, from predefined to random position assignments (Josip Habazin et al., 2017). The order picking activities include the retrieval of SKUs of interest and the picking of the right amount of products following the customer's demand (Josip Habazin et al., 2017). These activities can be manual or automatized, in the first case, the retrieval and the selection of the right amount of products is done by operators, whereas in the second case, the retrieving activity is performed by AS/RSs which transport the relevant SKUs to an area dedicated to picking (Josip Habazin et al., 2017). Picking activities can be performed in different ways, some requiring human presence (picker) and others entirely automated: picker to parts, parts to picker, robot to parts, parts to robot and picker less (Automated picking) (Yasmeen Jaghbeer et al., 2020).

Figure 2 represents a graphical classification of Order Picking Systems (Joo Ae Lee et al., 2015, Yasmeen Jaghbeer et al., 2020). Only the parts to picker and the robot to parts types will be analysed since they are the most relevant to the present study.

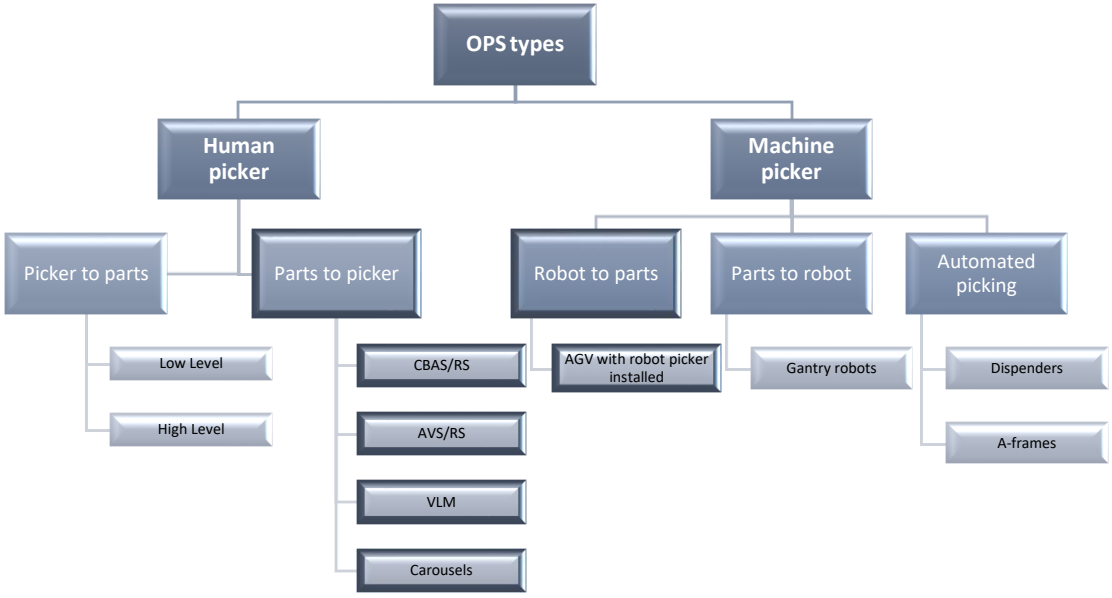


Figure 2: Graphical representation of different OPS Systems

The first AS/RSs ever introduced are called CBAS/RSs (crane-based automated storage and retrieval system) and they consist in automated cranes capable of performing simultaneously both horizontal and vertical movements along the aisles of the warehouse (Kaveh Azadeh et al., 2019). Crane based systems can be used both in single and multi-deep storages; to make the cranes easily work on a multi-deep storage, they are equipped with double-deep telescopic forks. (Kaveh Azadeh et al., 2019). For multi-deep storages, the crane system is supported by some conveyors that facilitate the storage and retrieval of pallets. If instead of pallets, the stored items are totes, the system is way more compact and it is called a mini-load automated storage and retrieval system (Kaveh Azadeh et al., 2019).

The automated system called AVS/RS offers even more flexible solutions compared to the crane based automated system. The AVS/RS allows to increase the overall throughput while decreasing the energy consumption since the shuttles used to store and retrieve pallets are much lighter compared to cranes (Giulia Bruno and D'Antonio, 2018). The system consists of shuttles that focus on horizontal movements and of lifts dedicated to vertical movements of pallets or shuttles. The AVS/RS system can be tier-captive or tier-to-

tier. In the first configuration, each tier has its own shuttle that focuses on storing or retrieving the pallet on that specific tier level. In the second configuration, the shuttle can move among different tiers by using the lift that allows vertical movements (Marchet G. et al., 2013). SBS/RS are part of the AVS/RS and they are generally used in mini-load warehouses (Lerher T. et al., 2015).

Another type of automated storage and retrieval systems are vertical and horizontal carousels which are mainly used for products with medium to small sizes. They are made of shelves that rotate either vertically or horizontally and they carry the relevant parts to the picker following the demand order (Kaveh Azadeh et al., 2019). The VLM (vertical lift modules) are similar to carousels and they are characterized by two columns of storage and a lifting crane in the middle that extracts the right SKUs and it brings them directly to the picker (Kaveh Azadeh et al., 2019).

The CBAS/RS, AVS/RS, VLM and Carousel fall into the category of those systems that bring parts to the picker in order to perform the picking activity (Yasmeen Jaghbeer et al., 2020).

In some cases, the order picking activity is performed by a robot, the robot travels towards the stock units and selects the relevant ones in the right amount. One example is given by the innovative picking system proposed by Nobutaka Kimura et al. (2015) where a robotized dual arm is mounted on an AGV to grant a flexible order picking for a high-mix warehouse (Nobutaka Kimura et al., 2015). Other robotized order picking solutions are the TORU and the SOTO robots. They are AGV able to pick items from the shelves, TORU can automatically pick cubic items without human assistance and SOTO directly handles totes (Richard Bormann et al., 2019).

2.2 Discrete Event Simulation in the context of automated warehouses for order picking activities

From the available past literature, it can be observed that many efforts have been put into finding ways to identify the best design configuration and to evaluate the performance of Automated Storage and Retrieval Systems using two main different approaches: Discrete

Event Simulation and development of analytical models in some cases tested and validated through simulation (Eder M, 2020).

The next sections of the present literature review will focus at first on the main studies supported entirely by a Discrete Event Simulation approach, it will then cite some studies supported by a combination of both DES and other approaches. The design/performance analysis presented in the following sections have been categorized based on the type of system studied. The performance studies of AS/RS which do not include the use of DES will not be considered. As already anticipated, the Discrete Event Simulation is often used in AV/RS literature, through this approach, the studied system is defined as a series of instantaneous occurrences or discrete events among whom the system is considered fixed (M.Law, 2015). It is important to underline that every study, even if they share the Discrete Event Simulation approach, they monitor different KPIs on different AV/RS.

2.2.1 DES in the context of AVS/RS

An example is given by the work written by Lerher T. et al. (2015) where discrete event simulation has been used to evaluate how the performance of a Shuttle Based Storage and Retrieval system, also called SBS/RS, part of the AV/RS, is affected and how throughput performance varies if rack configuration (number of tiers, number of aisles, number of columns and therefore length and heigh of storage racks), velocity of shuttles and lifts vary. The study concluded that the throughput capacity of the system depends on the throughput performance of the lift multiplied by the aisles, so if the number of tiers decrease and aisles increase, the throughput of the system should improve (Lerher T. et al., 2015). A later study by Lerher T. et al. (2016) tried to deepen the analysis on SBS/RS performance. In this article, a Discrete Event Simulation is applied to a SBS/RS System to evaluate its performance with specific focus on one KPI: Throughput. Nine different racks configurations have been studied and for each one of them the system's throughput capacity has been recorded. It was observed that the performance of the entire system is greatly dependent on the rack configuration (number of tiers, columns, and isles) and on the velocity of the shuttle/lift which however is limited by physical constraints. Because the study only focuses on throughput performance, it lacks considerations in terms of energy

consumption and energy regeneration which have some influence when decisions need to be made on the optimal design of the system (Lerher T. et al., 2016).

A similar study by Ekren B. et al. (2015) uses discrete event simulation and in which the main KPIs are the utilization of shuttles, utilization of lifts, cycle time of retrieval and cycle time of storage activity. The study gives some important contribution on warehouse design for tier captive SBS/RS systems and it identifies the best rack configuration for class-based storage policy through 10 different iteration of the simulation model. It is underlined that the study could be enriched even more if different arrival rates and different velocity profiles for shuttles and lifts are considered (Ekren B.Y. et al., 2015).

In other studies, cycle time seems to be the main element to describe the performance of an AVS/RS System as it is shown in the work of Eken B.Y. et al. (2011). In this study, discrete event simulation is employed to find the best combination of vehicles and lifts given pre-defined rack configurations. The performance measures used to evaluate each combination are cycle time, utilization of vehicle and utilization of lifts. It has been concluded that scenarios with larger number of lifts perform better compared to scenarios with lower number of lifts given that the number of vehicles is the same for both cases. It is important to underline that this study is not complete and exhaustive because it does not include considerations on costs (Ekren B. Y. and Heragu, 2011).

Marchet et al. (2013) introduced in their study some considerations on costs too. In this article discrete event simulation has been used to understand what the optimal rack configuration (number of aisles, tiers, and columns) for autonomous storage operated by AVS/RS could be. The rack configuration is very relevant as it has been underlined that the total annual costs related to autonomous warehouses are linked to vehicles, lifts, and space costs. During the simulation exercise, KPIs such as Throughput, flow time and cost have been monitored. The results suggest that the rack configuration has an impact on throughput and, based on how many tiers or how many columns the storage area has, different bottlenecks can be identified. In fact, if the storage develops on height (high number of tiers) the bottleneck may become the lift, so the throughput of the aisle is equal to the lift's throughput. On the other hand, if the storage system has longer aisles and less tiers, the bottleneck may become the vehicle and similarly to what happens for the

previous case, the throughput of one aisle will be equal to the throughput of the vehicle (Marchet G. et al., 2013).

Kriehn et al. (2018) used discrete event simulation to monitor changes in throughput in SBS/RS Systems when some specific storage management policies are applied contrary to performing a random storage assignment. The results of the study suggest that when class-based storage, sequencing of retrieval requests and warehouse reorganization are put in place either individually or, in some cases, in combination with each other, the system's throughput increases considerably. This leads to a reduction in processing time and lower energy consumption (Kriehn T. and Fittinghoff M., 2018).

A later study includes a focus on sustainability, and it has been written by Akpunar et al. (2017). In this study different warehouse configurations have been tested in order to select the one that guarantees the highest vehicles utilization and minimum energy consumption considering an AVS/RS System. Discrete event simulation has been implemented and 81 scenarios, which differ from each other in the rack configuration, have been analysed. It has been observed that energy consumption decreases when the warehouse is characterized by a low number of tiers and a high number of aisles. When the number of tiers or columns increase, energy consumption increases as well (Akpunar A. et al., 2017).

2.2.2 DES in the context of Crane based systems

Colla V. et al. (2010) used discrete event simulation to evaluate what could be the best storage policy in the context of crane based automated storage and retrieval systems. The storage policies evaluated are: FIFO reordering, Stacks reordering and Space reordering (Colla V. and Nastasi G., 2010). Different KPIs are monitored, the most important ones are listed below:

- Throughput
- Average Stock
- Receptivity (how many units the warehouse can stock)
- Handling potentiality (the average number of units handled by AV/RS)
- Fragmentation

Like the previous study, the paper of Vishwesh Singbal and K.Adil (2019) discusses how the performance of a crane based automated system changes based on the storage policy employed (Random or Across aisles full turnover). The study also evaluates the changes that happen if the number of aisles and the number of products vary. The study shows that the performance (evaluated in terms of Expected Travel Time Per Request) of the AS/RS is better when a random storage policy is employed no matter what is the number of aisles or the number of products taken into consideration (Vishwesh Singbal and K.Adil, 2019).

Lerher T et al. (2014) created a model that could support the activity of warehouse design when crane-based systems (mini load AS/RS) are employed. The study specifically focuses on energy consumption, the literature already written up to that point, extensively treated KPIs such as travel time, throughput, and cost. The model has been implemented through discrete event simulation and it has been observed that based on the variation of different factors (type of mini-load, velocity of devices), scenarios characterized by high velocity profiles resulted in higher CO2 consumption (Lerher T. et al., 2014).

2.2.3. DES in the context of VLM

A study conducted by Rosi B. et al. (2016) investigates how the performance of a single-tray VLM changes if the velocity profile of the lift and the dimensions of the VLM change. Through discrete event simulation, the study was able to compare the throughput of 4 different VLM configurations, and, it concluded that throughput improves when the height of VLM decreases and when the velocity of the lift increases (Rosi B et al., 2016).

Simulation is used by Battini D. et al. (2016) to determine how to improve throughput in a dual-tray VLM system. Given a specific set of VLM characteristics the simulation generates 10.000 random picking orders and different policies have been compared. It has been noted that Class Based Storage (storing most used items closer to the bay) and batch retrievals (higher probability to get more than one item on one tray) have influence in throughput improvements. The study also gives insights on how the operator can help the system performance by using a batch order picking approach (different orders are managed at the same time) (Daria Battini et al., 2016).

2.2.4 DES used to compare different AS/RS

Other studies are less selective and broader in their analysis. An example is given by the work created by Ekren et al. (2012) where the discrete event simulation approach is used to compare the performance of two well known AS/RS systems which are AVS/RS and CBAS/RS (Crane-based automated storage and retrieval system). In particular, the focus is put on tier-to-tier AVS/RS systems and aisle to aisle CBAS/RS. The simulation software used is ARENA and during this study, the experiment has been iterated 198 times in total (considering different values for number of vehicles, lifts, cranes, number of aisles, bays, tiers and two variants in demand). Contrary to the already mentioned studies, in this case the spectrum of KPIs is much wider. For each exercise 5 different KPIs have been monitored:

- Average Flow time
- S/R device average utilization
- Average waiting time in the S/R device queue
- Average number of jobs in queue waiting to be processed
- Cost

The outcome of the study showed that AVS/RS system performs better compared to CBAS/RS. The first ones on average are characterized by lower Flow time considering the same utilization for both S/R devices. Moreover, it is shown that AVS/RS have usually less jobs waiting in a queue and shorter waiting time in the queue compared to CBAS/RS. Although AVS/RS seems to be performing better compared to the other option, it is highlighted several times in the paper that AVS/RS is also the option that costs the most. Consequently, if there is the need to implement a material handling device at a lower cost, the option CBAS/RS would be the preferred one between the two (Ekren B. Y. and Heragu, 2012).

Bruno et al. (2016) decided to shift the focus of past studies by introducing in their article the concept of sustainability in the evaluation of AVS/RS performance, in this context the energy consumption of the system is considered as a KPI together with cycle time and devices utilization. The study tries to compare traditional crane-based systems with AVS/RS. A conceptual model is first developed and then it is implemented through Discrete

event Simulation. The results obtained show that AVS/RS, apart from giving benefits in terms of improved cycle time, it also provides considerable reduction in energy consumption (Bruno G. et al., 2016).

A more recent simulation-based analysis with focus on energy consumption is given by Guerrazzi E. et al. (2019) in the article “Energy Evaluation of Deep-Lane Autonomous Vehicle Storage and Retrieval System”. Through simulation, it has been observed that the utilization of AVS/RS can allow energy savings up to 60% compared to CBAS/RS (Guerrazzi E. et al., 2019).

2.2.5 Design and performance evaluation of AS/RS with other methodologies validated through DES

In some studies, simulation is used in combination with other approaches. A first example which is also one of the first studies linked to AVS/RS performance evaluation is given by the work of Malmborg (2002) where an analytical model has been developed to estimate AVS/RS performance in terms of cycle time and vehicle utilization under different rack configurations. The study aims at comparing AVS/RS with AV/RS performance. The analytical model is then validated through simulation (Malmborg Charles J., 2002).

Eder M. and Kartnig G. (2016) later developed an analytical model to determine the best rack configuration and geometry capable to achieve the greatest throughput for a S/R Shuttle System. Discrete Event simulation through SIMIO simulation software is used in this study to validate the analytical model. The analytical model suggests that as the height and length of the racks increase, the throughput at first improves up to a certain point after which it tends to worsen (Eder M. and Kartnig G., 2016).

A more recent study by Lerher T. et al. (2020) introduces in the already existing literature a focus on the performance evaluation of an AVS/RS with multiple-tier vehicles. An analytical model is able to calculate cycle time and throughput of this innovative storage system and a discrete event simulation validates the results obtained analytically. This study could be particularly beneficial during the system design process as it gives valuable insights on the best rack configuration and velocity profile of devices (Lerher T. et al., 2020).

Lerher T. (2017) used DES together with Design of Experiments (DoE) applied to SBS/RS in order to better identify the optimal performance in terms of throughput. From the study it has been concluded that different factors affect throughput: number of columns, velocity and acceleration/deceleration of both shuttles and lifts. The study let these factors vary to see what the final throughput would be and, it has been observed that the best scenario is the one with the smallest number of columns and the greatest number of tiers (Lerher T., 2017).

The work of Sgarbossa et al. (2019) focuses on the throughput evaluation of a VLM System under different storage policies. In the paper it is stated that the overall performance of this system is tightly dependent on the performance of the operator and the dual bay VLM. The study focuses on the case in which the VLM is the bottleneck and it concluded that one way to improve the throughput of the system in this situation would be to switch from a random storage policy to a CBS (Class-Based Storage) one. The study develops an analytical model that is then validated through Discrete Event Simulation (Fabio Sgarbossa et al., 2019).

The performance of a carousel system is analysed in the work of Jennifer A. et al. (2012). They developed an analytical model for cycle time, supported by Discrete Event Simulation. The aim of the paper was to determine what is the retrieval policy applied to carousels that could increase its throughput. The results show that a batch processing policy allows to increase throughput compared to sequential processing, as a consequence, if batch processing is applied, financial gains can be achieved, as a lower number of carousels can be installed to achieve satisfactory results in terms of throughput (Jennifer A et al., 2012).

It has been noted that there are no articles that use the Discrete Event Simulation to evaluate the performance of a robot to parts system. Several studies use other methodologies, like mathematical models, to estimate cost, lead time or flexibility of these OPS (Yasmeen Jaghbeer et al., 2020).

The following table summarizes the studies cited in this literature review.

Table 1: Literature review on AS/RS performance analysis and AS/RS design criteria

Year	Author/s	System studied	Characteristics	Main KPIs	Model
DES in the context of AVS/RS					
2015	Lerher T. et al.	SBS/RS	Tier-captive DCC	Throughput	Simulation
2016	Lerher T. et al.	SBS/RS	Tier-captive SCC/DCC	Throughput	Simulation
2015	Ekren B. et al.	SBS/RS	Tier-captive DCC	Utilization Cycle time	Simulation
2011	Ekren B. et al.	AVS/RS	Tier to tier	Utilization Cycle time	Simulation
2013	Marchet G. et al.	AVS/RS	Tier captive SCC	Throughput Flow time Cost	Simulation
2018	Kriehn T. et al.	SBS/RS	Tier-captive SCC/DCC	Throughput	Simulation
2017	Akpunar et al.	AVS/RS	Tier to tier DCC	Energy consumption	Simulation
DES in the context of Crane based systems					
2010	Colla V et al.	Crane based system	-	Throughput Average Stock Receptivity Handling potentiality Fragmentation	Simulation
2019	Vishwesh S. and Gajendra K.Adil	Crane-based system	Single-crane multi-aisles	Expected Travel Time	Simulation
2014	Lerher T. et al.	Mini-load AS/RS	SCC/DCC	Energy consumption	Simulation
DES in the context of Carousels and VLM					
2016	Rosi B. et al.	VLM	Single-Tray	Throughput	Simulation
2016	Battini D. et el.	VLM	Dual-Tray	Throughput	Simulation
DES used to compare different AS/RS					
2012	Ekren B et al.	CBAS/RS vs AVS/RS	Tier to tier	Flow time Utilisation Waiting time N° jobs waiting Cost	Simulation
2016	Bruno G. et al.	CBAS/RS vs AVS/RS	Tier to tier SCC/DCC	Cycle time Utilization Energy consumption	Simulation
2019	Guerrazzi E. et al.	CBAS/RS vs AVS/RS	Tier captive SCC	Energy consumption	Simulation
Design and Performance evaluation of AS/RS with other methodologies in combination with DES					
2002	Malmborg Charles J.	AVS/RS	Tier to Tier SCC/DCC	Cycle time, devices utilization	Analytical model Simulation
2016	Eder M. et al.	S/R shuttle system	Tier captive	Throughput	Analytical model Simulation

2020	Lerher T et al.	AVS/RS	Multiple tier shuttles SCC/DCC	Cycle time Throughput	Analytical model Simulation
2017	Lerher T.	SBS/RS	Tier-captive SCC/DCC	Throughput	Simulation DOE
2019	Sgarbossa et al.	VLM	Dual bay	Throughput	Analytical model Simulation
2012	Jennifer A. et al.	Carousel	Horizontal	Throughput	Analytical model Simulation

3. FlexSim: Simulation Software

FlexSim is a simulation software that allows to build both 3D and 2D simulations of a specific system. The flexibility offered by the software gives the possibility to build a digital model that replicates in the most accurate way a real-life system. This software can be used in several different ways to simulate different processes, from manufacturing processes to warehouses and healthcare systems. It can be helpful in several situations, for instance when the end goal is to evaluate how a change in warehouse layout or how the introduction of new elements in the process can impact the overall system. The software offers, in fact, the possibility to gather and show data in an organized way and it enables to run and compare different simulations which differ from each other in terms of input parameters giving the possibility to identify the most convenient one.

FlexSim is considered as a way to test and optimize a process before applying the changes in a real setting which would imply high-cost investments at a high risk. The software lets the user work in a risk-free environment and it give insights on how to fix bottlenecks, how to allocate resources in the most optimal way, how to improve throughput, how to reduce queues and waiting times (FlexSim).

Some of the main functionalities, that will be mentioned in the following chapters in relation to the simulation model built for this thesis, are 3D model, Process Flow, Dashboard, Experimenter and Optimizer. These elements will be briefly described in the sections below.

3.1 Working with a 3D model

A 3D model helps in better visualizing the space that the process will occupy if implemented in real life. It can be seen as a more accurate way to observe the path that the resources involved will follow and how the system behaves in general. FlexSim offers an Object-oriented simulation through a system of classes and sub-classes. The user interface of a 3D model can be observed in Figure 3.

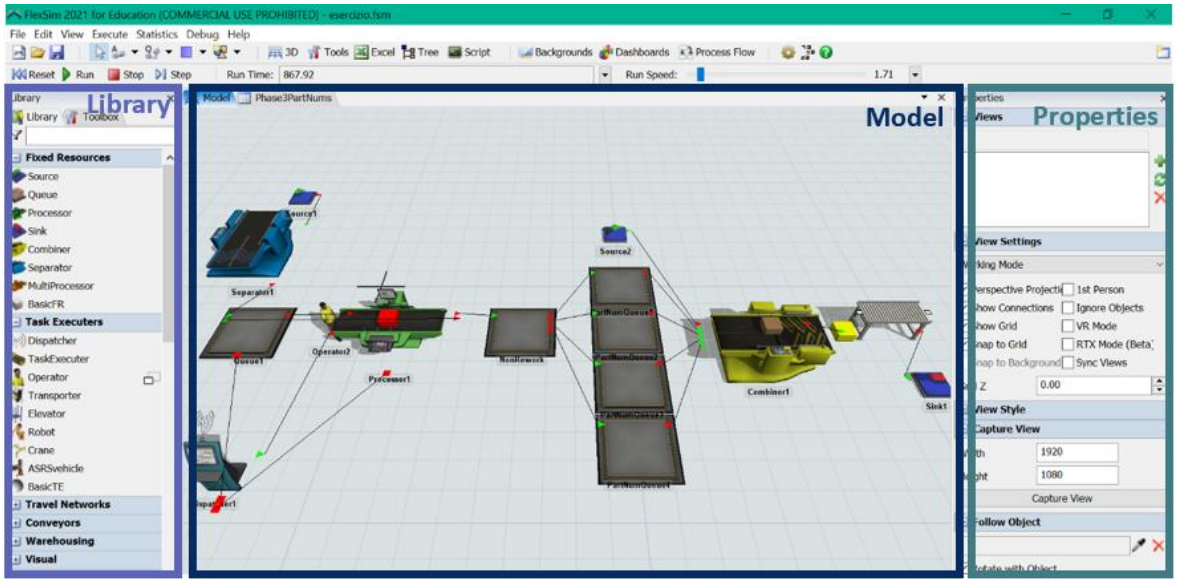


Figure 3: FlexSim's 3D model User Interface

From Figure 3 it is possible to identify three distinctive areas: Library, 3D Model and Properties.

From the Library it is possible to insert in the model different objects, created from classes, through a *Drag and Drop* action. Some examples of objects relevant to this study and offered by the software are Fixed Resources, Task Executers, Warehousing and AGV (FlexSim). Another type of object that is not directly offered in the library is the Flow Item.

- Flow Item: Items that flow into the process and during the simulation, they can be transformed, they can move, they can carry information stored in *labels* (static or dynamic) attached to them. Flow Items can be boxes, pallets, totes, spheres but also people, orders, trucks depending on the specific simulation. In the simulation

model described in the following chapters, an example of Flow Item is a pallet containing boxes.

- Fixed Resources: Objects that interact with the Flow Items processed in the model and they are static meaning that they do not move during the simulation. As an example, they can be sources which generate items, processors that process items, buffers, combiners, separators.
- Task Executers: Objects in the form of operators or machines that can move around the model and they can interact with the Flow Items. In the simulation model that will be described in the following chapters, the Task Executers are the AGV, the satellites and the robots which stock, retrieve pallets and satisfy orders.
- Warehousing: These objects allow to put in place the rack configuration that most mirrors the real-life layout. FlexSim offers different types of storage systems, that can be highly personalized, such as standard racks, push backs and gravity racks, drive in racks. The latter will be used in the simulation model built for this study.
- AGV: They are a specific type of task executers connected to an AGV travel network. They can be used to transport and carry Items in the simulation model. They are extremely relevant in our study since the technology that the thesis aims at evaluating can be represented by an AGV in the model.

3.2 Working with a Process Flow

The Process Flow is a great tool accessible from the main FlexSim window that allows to build and structure the model's logic through flow charts. The users can insert activities in the Process Flow that will mirror the real activities happening during the process. Once the users run the Process Flow, tokens will be created (green circles in Figure 4) and they will follow the flow specified in the chart, it is possible to link tokens to specific items generated in the 3D model so that there is a direct link between the 3D and the 2D model. An example of Process Flow can be observed in Figure 4.

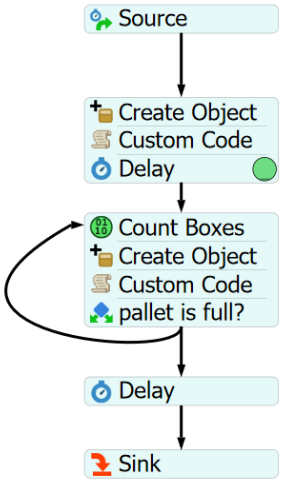


Figure 4: FlexSim's Process Flow example

3.3 Working with Dashboards

Dashboards are blank pages that can be populated by the user with different types of information, mainly graphs that display in real time relevant data while the simulation is running. As an example, the charts can display information about the throughput of a specific machine, the processing time, the state of machines, the work in progress. In addition to that, the user has the freedom to choose the most suitable type of chart, from pie charts to bar charts and time plots. An example of dashboard can be observed in Figure 5.

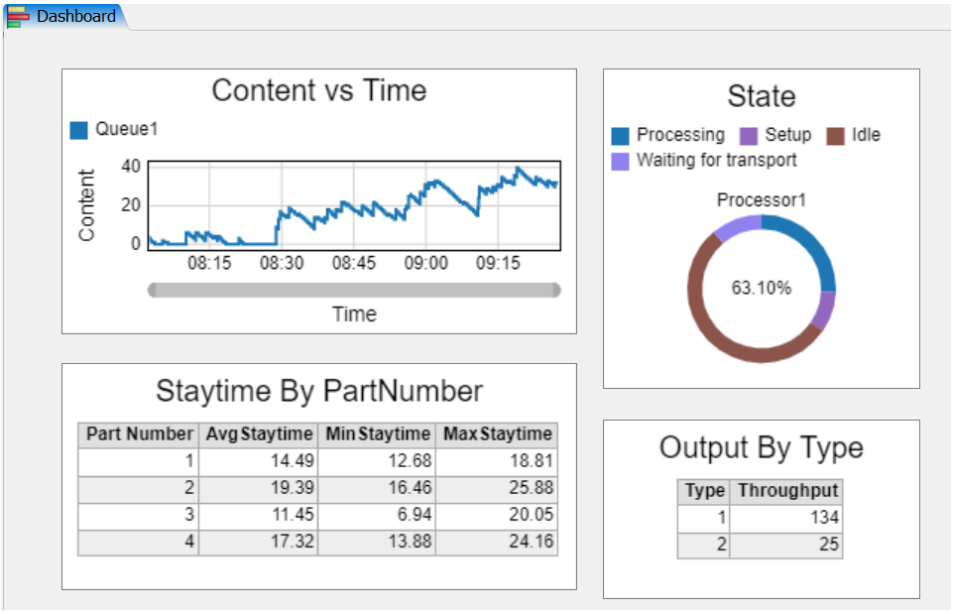


Figure 5: FlexSim's Dashboard example

3.4 Working with the Experimenter and the Optimizer

The Experimenter and the Optimizer are two important tools that help the users understand what the best variable setting for their model is. With the Experimenter it is possible to run the same simulation several time, each time changing some variables to see how those variables impact the process performance. The Optimizer tool will then try to understand what the best variables values are, in order to maximize the relevant performance measures.

4. Warehouse storage policies and operating logic

Before explaining in more details the structure of the model developed for this study, it is important to explain how the storage policies, that will be implemented, work. The model will focus on the implementation and comparison of five different storage policies to see which one could give the highest benefits when combined with the new technology. The examined storage policies in the context of deep-lane warehouses are: Random Storage, Class-Based Storage, Dedicates Slots Storage, Storage by Weight and Storage by Association Rules.

4.1 Random Storage

In this storage policy the warehouse slots are filled with a random approach meaning that every SKUs have equal probability of occupying available slots (Zaerpour N. et al., 2013). Figure 6 shows the Random storage policy.

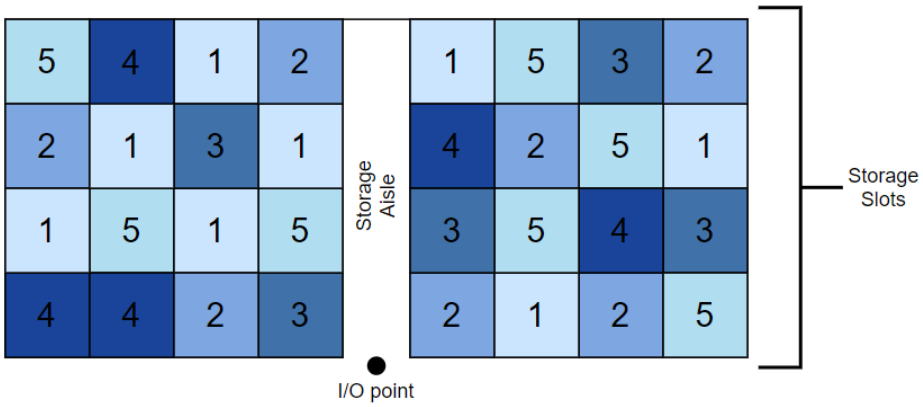


Figure 6: Warehouse with random Storage Policy

4.2 Class-Based Storage

Class-Based Storage policy allows to classify the SKUs into three different classes A, B, C which will be stored in dedicated areas in the warehouse (Area A, Area B, Area C). Class-Based Storage can be developed differentiating SKUs by using different product characteristics including picking frequency, products volume, quantity of products sold, values of product sales (Lorenc A. and Lerher T., 2019). The division is put into place based on the product involvement, generally the products which are highly involved (Class A) will account for 80%, product with medium involvement (Class B) will account for 15% and the remaining with low involvement (Class C) represent 5% (Lorenc A. and Lerher T., 2019). In the literature it is possible to find different percentages to the ones written above based on the specific study.

As an example, if the products are differentiated by number of products sold, the products with the highest number of sold products will be part of Class A, the products which are characterized by a low number of units sold will be part of Class C and all the remaining will be part of Class B. The items in class A are responsible for 80% of all products sold by the company, the items in class B are responsible for 15% of total sales and the ones in class C represent 5% of total sales. Generally, the items in Class A are a few but they are highly involved, on the other hand, products in class C are numerous but their level of involvement is low. Figure 7 shows the ABC classification through a graphical representation.

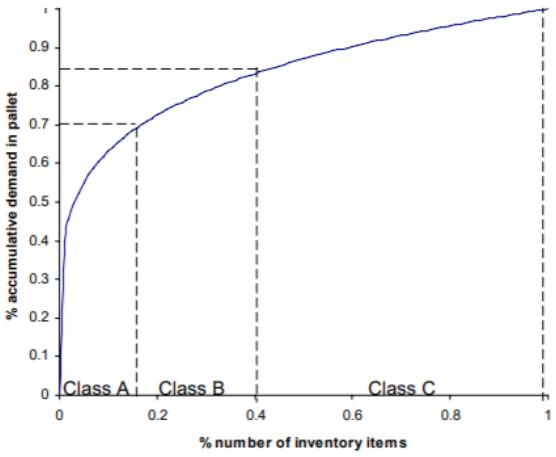


Figure 7: ABC curve (Yugang Yu et al., 2015)

After the classification, it is important to trace the different areas in the warehouse. The products in Class A will be stored closer to the depot point to guarantee an easy and quick access, the other two Classes will be stored further away from the depot point since they are characterized by a lower demand and they will be accessed less frequently. As Figure 8 shows, the shape and the location of the different areas change based on the organization of the warehouse and based on where the depot point is located. Different Areas configuration can be highlighted: Diagonal, Within Aisles and Across Aisles. All these configurations show that the Class A (black squares) is close to the depot point, the Class B (grey squares) is located in the middle of the warehouse a bit further away from the depot point and the Class C (white squares) is at the end of the warehouse.

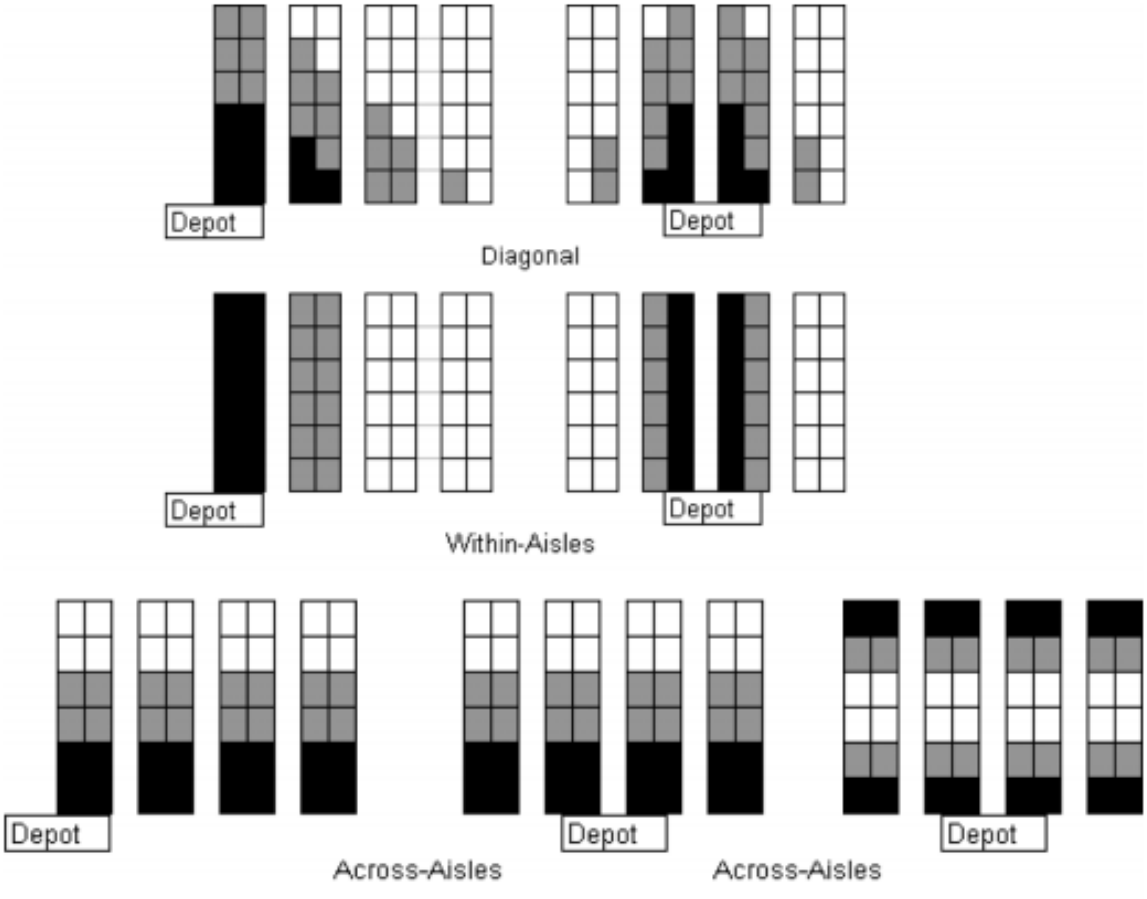


Figure 8: ABC Areas in a Warehouse (Aurelija., 2010)

When talking about Class-Based storage, the present study will be referring to picking frequency classification, which considers how many times the SKUs are appearing in the client orders without accounting for the quantity requested for each SKU. The percentages that will be used are equal to 60% for Class A, 30% for Class B and 10% for Class C.

4.3 Dedicated Slots storage

In a dedicated Slots Storage, the slots are assigned to specific products, even if the product is out of stock that place is only meant to be filled with that type of item (Aurelija., 2010). Figure 9 shows an example of Dedicates Slots Storage policy.

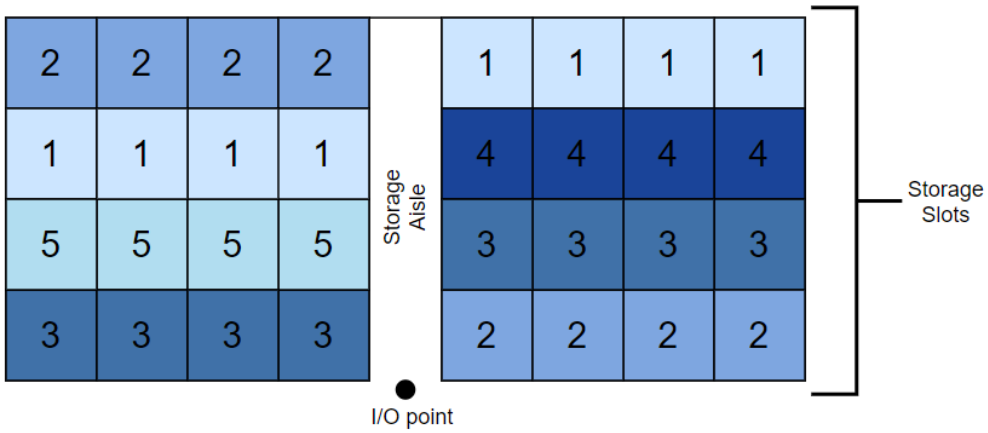


Figure 9: Warehouse Dedicated Slots Policy

4.4 Storage by weight

In this storage policy weight is the product characteristics that helps determining where the SKUs should be stored. It can be seen as one of the variants of the Class-Based storage where the characteristic of interest is the weight (Lorenc A. and Lerher T., 2019). In the variant “storage by weight” part of model that will be described in the following chapters, the SKUs have not been divided into classes, but the heaviest units have been stored close to the pallet retrieval point and the lightest products have been stored far away from the same point to see whether a storage like this one could give some additional benefits in terms of energy savings.

4.5 Storage by Association rules

Some studies have highlighted the importance of introducing data mining techniques, in particular the *Apriori* Algorithm, in the context of automated warehouses to improve the slot assignment activity so that the travel distance during the operations of order picking is minimized (Hau Ling Chan and King Wah Pang, 2011).

The *Apriori* Algorithm helps in finding hidden patters in a list of several picking orders, it identifies what combination of items is requested with the highest frequency through an iterative process (Rifki Fahrial Zainal and Fardanto Setyatama, 2016). The steps of the *Apriori* Algorithm are listed below (Online, 2021) and the algorithm's process flow is shown in Figure 10.

Step 1: The list of picking orders is taken as input. In this first step a 1 list item called C1 is created in order to show the frequency of appearance of that item in the orders. If the frequency of appearance of items in list C1 is higher than a certain threshold called *minimum support,* then the item is significant and it can be included in another list of items called L1.

Step2: Each item from list L1 is paired with another item from the same list in order to create subsets of items and their frequency is calculated by analysing the initial list of orders and stored in list C2. The subsets of items that have a frequency higher than the minimum support will be saved in list L2.

Step3: The subsets from list L2 are paired with other subsets in order to create a subset of three items, they are saved in list C3 and their frequency is calculated from the initial list of orders. If the subsets of items have a frequency higher than the minimum support, they will be saved in list L3.

Step 4: The algorithm continues until the most frequent itemset is found.

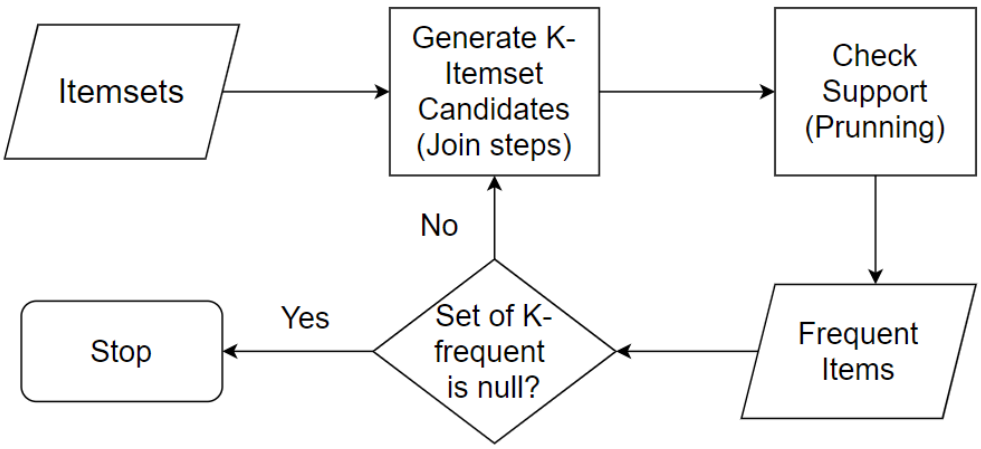


Figure 10: *Apriori* Algorithm Process Flow (Bagui, 2019)

The following numerical exercise exemplifies how the *Apriori* Algorithm works assuming that the total number of different products is 5 and the minimum support is 5. Each order indicates the type of products requested, a combination of products 1,2,3,4, and 5.

Initial picking orders
1-3-5
1-2
4-5-1
1-2-3-4-5
1-2-5
2-5-4
1-3-5
4-2-3

C1		L1	
Items	**Frequency**	**Items**	**Frequency**
1	6	1	6
2	5	2	5
3	4	5	6
4	4		
5	6		

In List C1 the frequency of appearance of the single items is recorded. For instance, item type 1 has been requested 6 times in the input order list. Only the items that have been requested more than 5 times (5 is the minimum support) will be part of List L1.

C2		L2	
Itemset	**Frequency**	**Itemset**	**Frequency**
1-2	3	1-5	5
1-5	5		
2-5	3		

From L1 a combination of items is listed creating all possible item sets. The frequency of appearance of the item sets is recorded in C2. The item sets that are requested less than 5 times are deleted and only the ones who are requested more than 5 times are saved in L2.

The example shows that in this specific case, Items 1 and 5 create the itemset that is most frequently requested by customers and therefore it is sensible to store items 1 and 5 close together.

4.6 Warehouse Operating logic

The present section describes in detail the conceptual model of the automated warehouse that will be studied in the following chapters through the simulation model: an automated deep-lane warehouse served by tier to tier vehicles each composed by a Shuttle, a Satellite and a Robotic Arm (Xv,Yv,Zv) aimed at satisfying picking orders. In order to describe the operating logic of the warehouse in the most complete way, different warehouse activities will be described: from customer order satisfaction to slots replenishment. In other words, the descriptions will focus on how the Picking order mission, Pallet retrieval mission and Pallet storage mission work. The coordinates of the pallets/slot to retrieve/reach is indicated with Xp,Yp,Zp whereas the coordinate of the lift is Zl.

4.6.1 Picking order mission

The behaviour of the warehouse during the picking order system starts with the arrival of customer orders that require the creation of mixed pallets that mirrors the expectations of the clients. As soon as the order arrives, the request waits for a vehicle (intended in this context as shuttle, Satellites and Robotic arm grouped together) to proceed with the picking activity. If a vehicle is available, the task to fulfil the order is assigned to that vehicle, otherwise, the order waits. When a vehicle is ready to fulfil the order, the latter is read by the System and the first type of product to include in the mixed pallet is identified. The shuttle travels horizontally with the aim to reach the slot where it can find the right type of product. The location of the different pallets of products in the slots can vary based on the storage policy that the warehouse is employing. If the slot is on the ground floor, the shuttle directly travels to destination, if not, the shuttle needs to wait for the elevator to reach the ground floor and to lift the vehicle at the right level. If the slot is occupied by another vehicle for another picking activity, the vehicle that needs to reach that slot has to wait for it to be freed. Once the vehicle finally reaches the slot, the satellite travels to the pallet, loads the pallet and brings it closer to the shuttle, so that the Robotic Arm mounted on the shuttle can pick the right number of SKU units requested by the customer. Once this activity is done, and the satellite has put the pallet back in the slot, the system analyses the order to see if there are other types of products that need to be picked. If so, the shuttle needs to travel to the right slot destination as previously explained, if there are no other types of

products to be picked, the shuttle travels towards the discharge area, it unloads the mixed pallet and it waits in the buffer for a new order assignment. The Activity diagram that explains the logic behind the picking activity is shown in Figure 11.

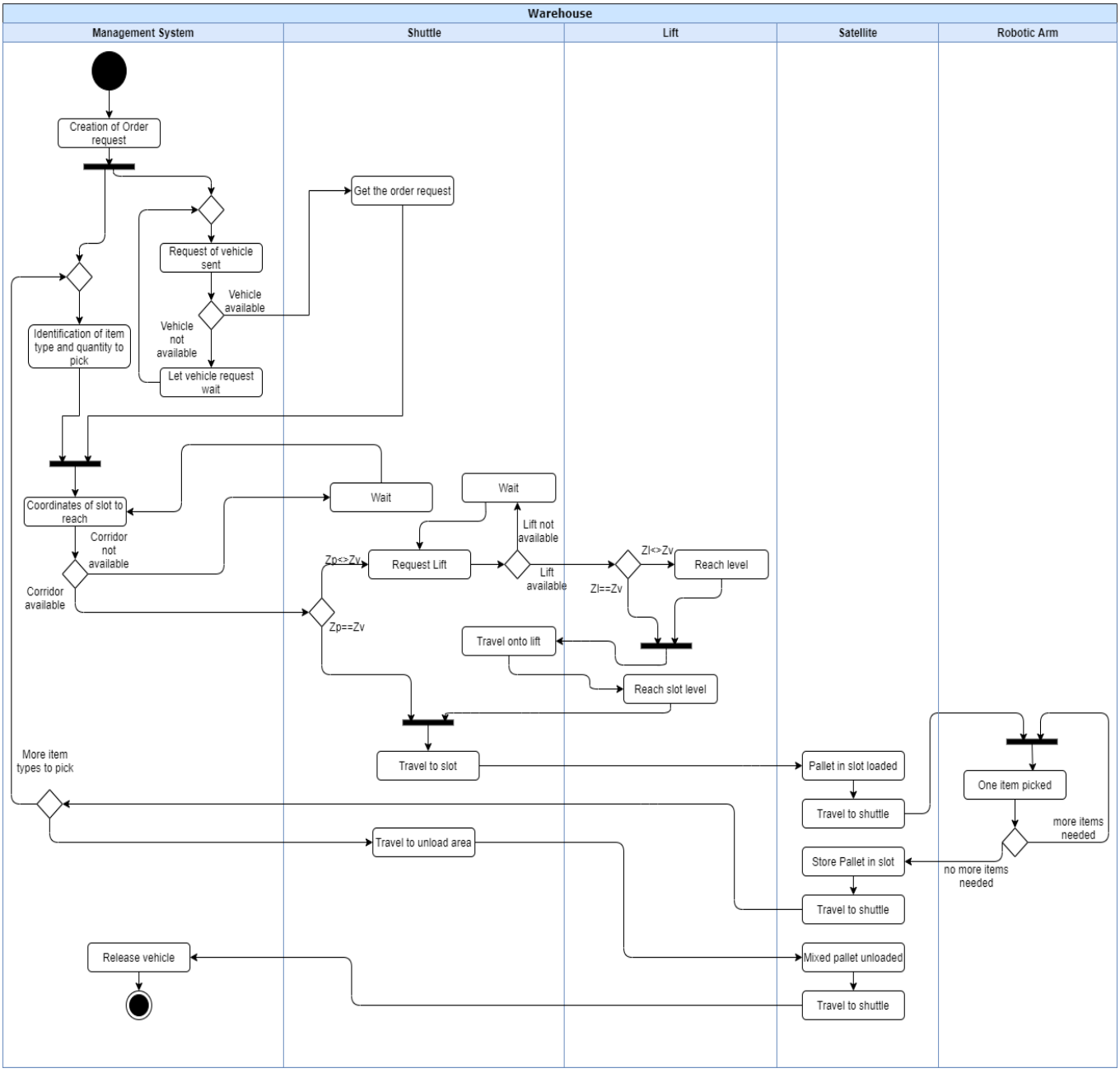


Figure 11: Order Picking Activity – Activity Diagram

4.6.2 Retrieval mission

In the Retrieval missions the aim is to take out of the warehouses entire pallets filled with the SKU requested by the client. The Activity Diagram that explains how the retrieval missions work is shown in Figure 12. In this case, similarly to the previous scenario, the transaction that indicates the need for a retrieval mission enters the system and it is put on hold until a vehicle is ready to process that specific order. Once the vehicle is available, the system associates that vehicle to the order, and it tries to locate the pallet to retrieve giving to the vehicle the pallet coordinates to reach (Xp,Yp,Zp). The location of the pallet will vary based on the storage policy that is in use in that moment. The vehicle travels towards the destination taking the lift if the pallet slot is situated on a different level compared to where the vehicle is located. Once the vehicle is in front of the right slot, the satellite detaches from the vehicle and loads the pallet of interest onto the vehicle.

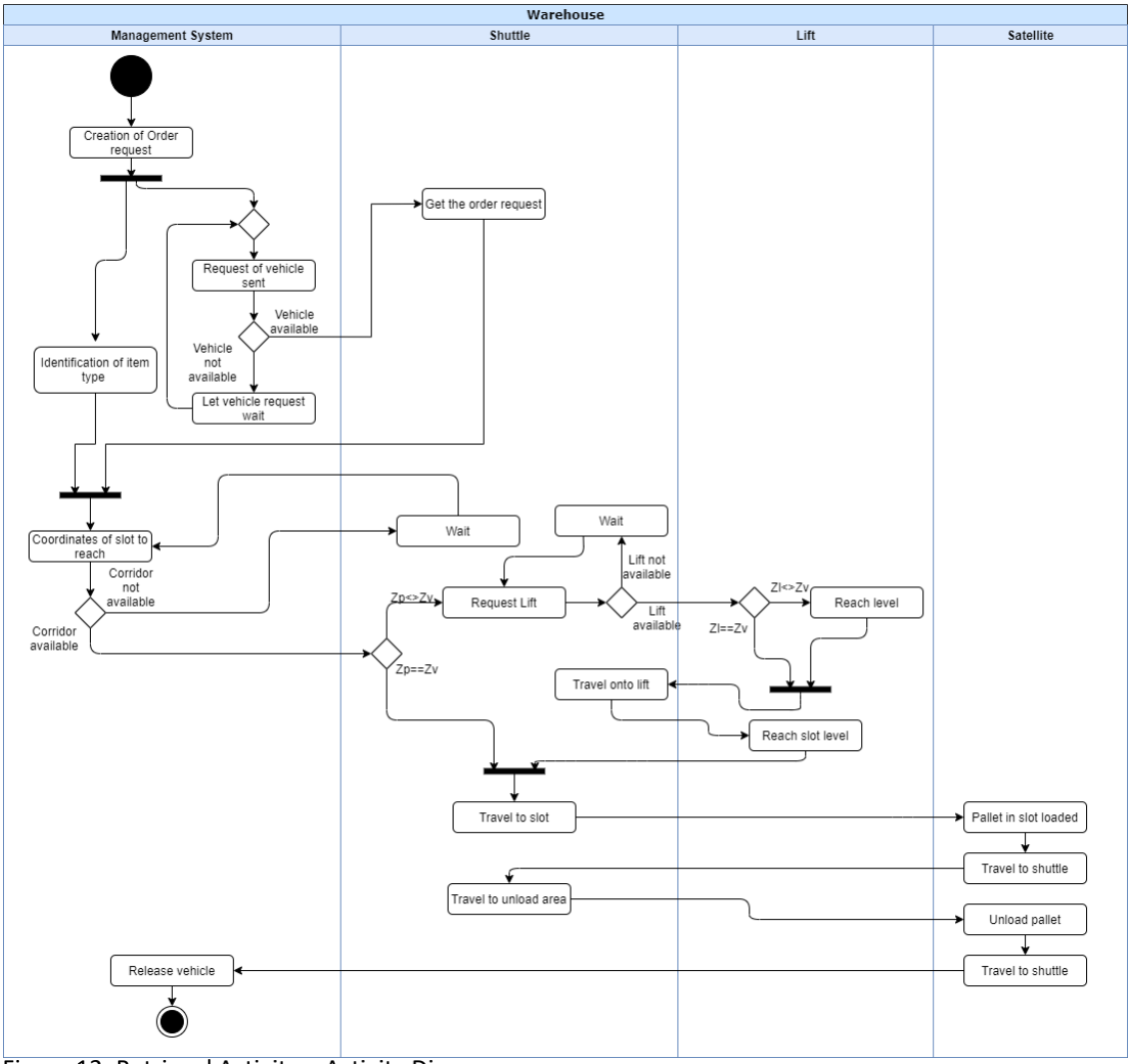


Figure 12: Retrieval Activity – Activity Diagram

The AGV can now reach the discharge area and the entire vehicle is released and it will wait for a new order.

4.6.3 Storage mission

When the storage activity is required, the system identifies what the optimal location to unload the incoming pallet is. If there is vehicle availability, the storage mission is assigned to a vehicle. The system gives the vehicle the coordinates of the buffer where the pallet to load is located. Once the pallet is loaded, the vehicle is given the coordinates that indicate a position inside the rack, where the pallet should be stored. The vehicle reaches the destination by using the lift if necessary. Once the AGV reaches the right slot, the Satellite detaches from the shuttle and it unloads the pallet in the right position. The entire vehicle will then travel back to its buffer position waiting to perform a new activity. The Activity Diagram that explains how the storage missions work is shown in Figure 13.

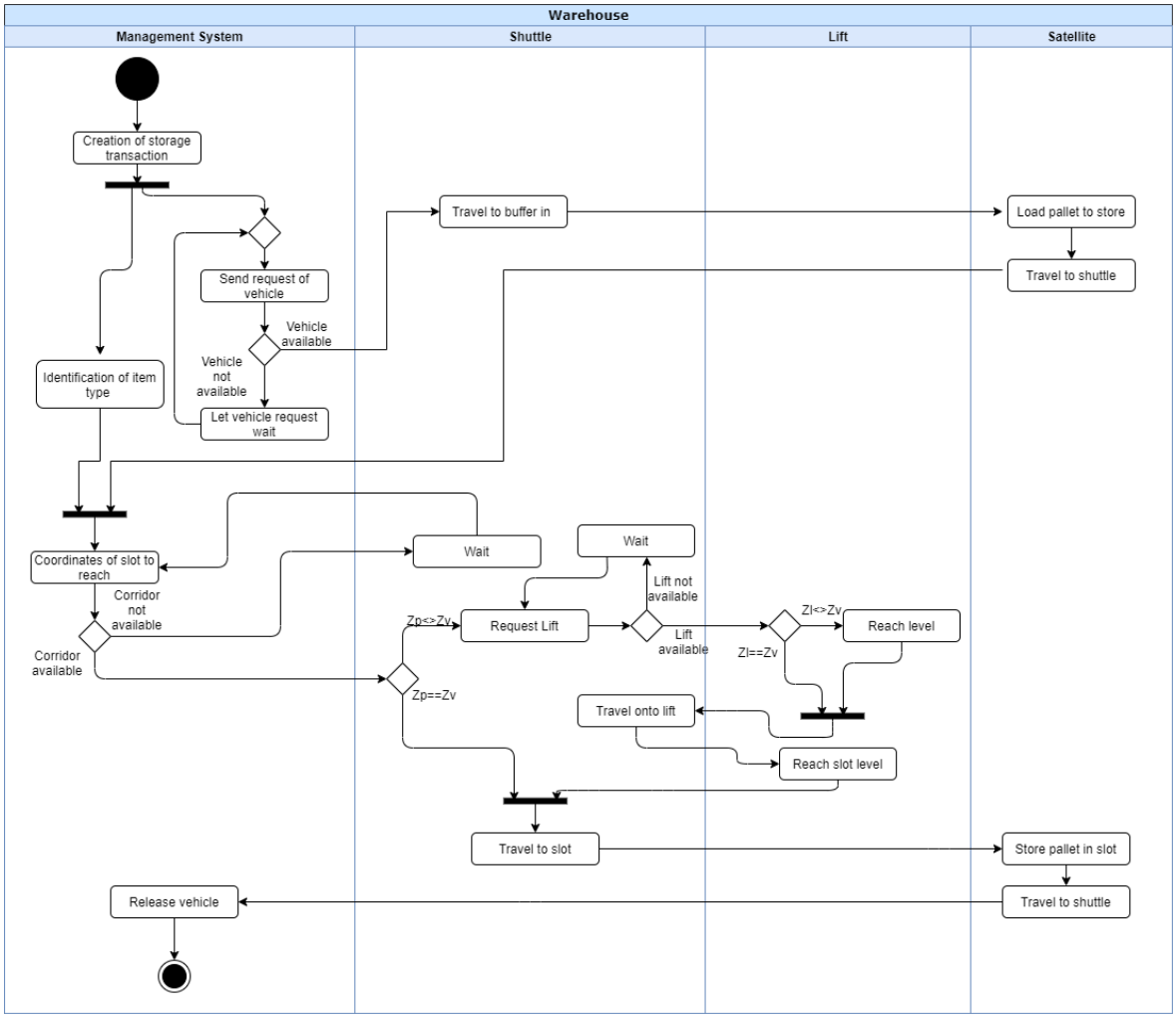


Figure 13: Storage Activity – Activity Diagram

5. Implementation of Simulation model

The simulation model built in FlexSim for this study aims at representing a generic deep-lane warehouse served by automatic vehicles which consist of a shuttle that is mainly used for travels on the x and y axis, a satellite which mainly travels on the x axis and a robot which is activated during the picking operations. The movements on the z axis are provided by lifts. The vehicles operate in the warehouse following a FIFO (First In First Out approach) and it is assumed that the warehouse can receive mixed orders as well as simple retrieval orders. The simulation model will not include the storage activities (replenishment of pallet slots) as it is not part of the focus of this study. The efficiency of the warehouse will be evaluated under different scenarios based on variations concerning its rack configuration, number of vehicles, frequency of incoming orders, number of SKUs and type of Storage Policy used. Figures 14 and 15 show the main elements that make up the generic physical structure of the facility. As it is shown below, it is possible to distinguish different design elements like corridors, bays, levels, vehicles.

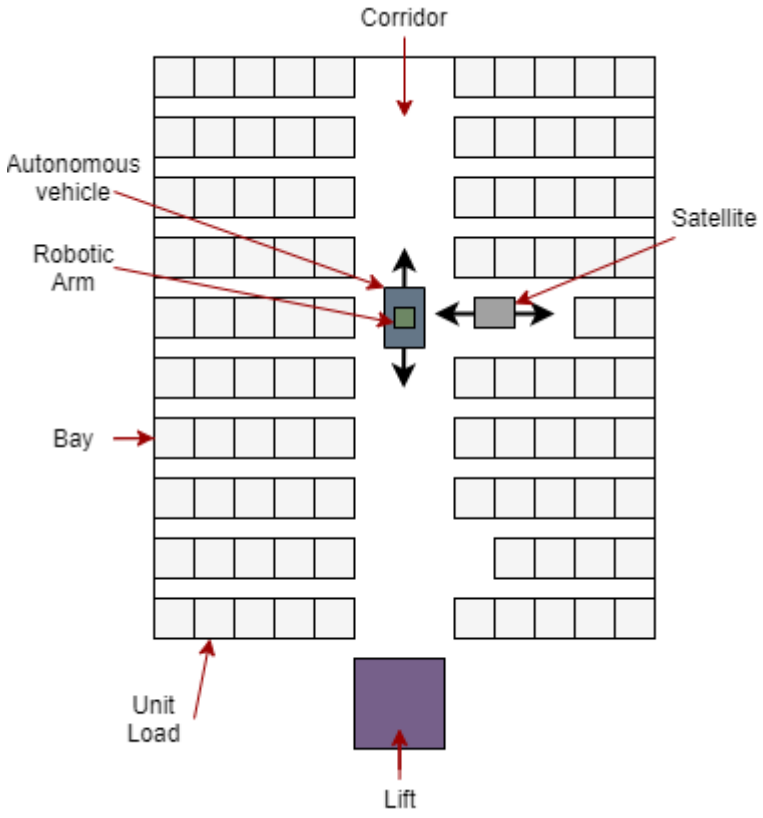


Figure 14: View from above of warehouse and its main elements

Some of these elements will vary during the simulations to register how they affect the overall warehouse performance and consequently the performance of the new technology.

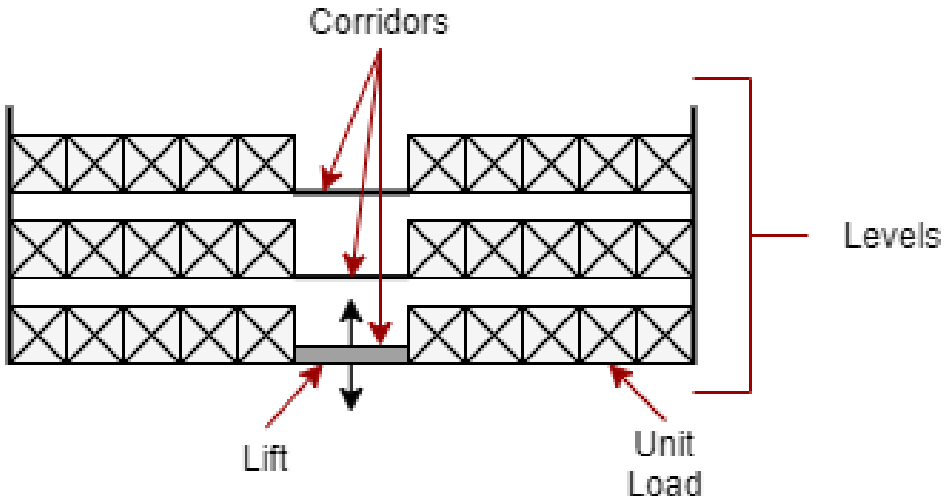


Figure 15: Frontal view of warehouse and its main elements

5.1 Key input Model Parameters and simulation design

As already anticipated, this study will perform several simulation runs which will differ from each other based on the value of pre-determined parameters. Some key parameters have been identified and they will be varied during different simulations to see what their effect on the warehouse performance is. The input model parameters that will be taken into consideration are the frequency of order arrival, the number of corridors, the number of levels, the number of available vehicles, the number of SKUs and the type of storage policy. The aim is to identify what is the best storage policy for picking operations under different circumstances. The simulation will be divided into two scenarios. In the first scenario the study wants to identify the best storage policy under different circumstances given the fact that the warehouse can accept as input orders both picking orders and retrieval orders. In the second scenario the element of disturb will be removed, in this case the warehouse can only accept picking orders. The second scenario is a simplification of what happens in real life but it can be interesting to see if the isolation of picking orders leads to different results. With these two scenarios it will be possible to understand how the different storage policies react if the activities are not solely focused on fulfil mixed orders. The different parameters mentioned above are summarized in Table 2 where it is possible to see the

different values assigned to them in different simulation runs. For every parameter configuration, a total of five simulation runs will be performed, every simulation will be characterized by the same input parameter values but certain elements, like the sequence at which orders arrive, will vary since this information is generated through a random distribution that gives different outputs at every simulation run. The final result will be calculated as the average of the five simulation runs. This will guarantees a more precise and reliable result.

Table 2: Parameter input values

Parameter name (factors)	Parameter values (levels)
Number of corridors	2
	4
Number of levels	3
	6
Number of vehicles	3
	6
Number of SKUs	5
	9
Inter-arrival time of piking orders [s]	50
	100
Storage Policy	(1) Random
	(2) Class Based Storage
	(3) Dedicated Slots Storage
	(4) Storage by SKU weight
	(5) Storage by Association rules

Taking into account only one scenario, for every storage policy, different simulations will be performed, where every simulation is the result of the average of five simulations with the same input parameters. In order to evaluate the behaviour of the different storage policies under different circumstances, the study will use the full factorial design at 2-levels, meaning that for every storage policy there will be five factors and two levels (two possible values for each factor). Following the full factorial design approach, the total number of simulations that will be performed is equal to $levels^{factors}$, which translates into $2^{factors}$ in a 2-levels full factorial design (Jiju A., 2014). In this specific study, for every storage policy there will be $2^5 = 32$ simulations as it is possible to observe in Table 3. Since each

simulation is an average of other five simulations as already mentioned above, the actual number of simulations for each storage policy will be equal to $32 \times 5 = 160$.

Considering five different storage policies, the total number of simulations that will be carried out in one scenario will be equal to $160 \times 5 = 800$.

Table 3: Simulations that need to be performed for each storage policy

		factors				
N° Simulations	N° Runs	N° Corridors	N° Levels	N° Vehicles	N° SKUs	Orders Inter-arrival time
1	5	2	3	3	5	50
2	5	2	3	3	5	100
3	5	2	3	3	9	50
4	5	2	3	3	9	100
5	5	2	3	6	5	50
6	5	2	3	6	5	100
7	5	2	3	6	9	50
8	5	2	3	6	9	100
9	5	2	6	3	5	50
10	5	2	6	3	5	100
11	5	2	6	3	9	50
12	5	2	6	3	9	100
13	5	2	6	6	5	50
14	5	2	6	6	5	100
15	5	2	6	6	9	50
16	5	2	6	6	9	100
17	5	4	3	3	5	50
18	5	4	3	3	5	100
19	5	4	3	3	9	50
20	5	4	3	3	9	100
21	5	4	3	6	5	50
22	5	4	3	6	5	100
23	5	4	3	6	9	50
24	5	4	3	6	9	100
25	5	4	6	3	5	50
26	5	4	6	3	5	100
27	5	4	6	3	9	50
28	5	4	6	3	9	100
29	5	4	6	6	5	50
30	5	4	6	6	5	100
31	5	4	6	6	9	50
32	5	4	6	6	9	100

The other scenario will experience the same number of simulations. As a result, the total number of simulations that will be performed, considering both scenarios will be equal to 1600.

Through this extensive study it will be possible to not only compare the single storage policies based on different factor configurations, but it will also be possible to determine what the best storage policy is as the varying factors change.

5.2 Key Performance variables

All simulations will be evaluated based on specific performance variables which act as the output of the study. The final discussion of the results will be based on the numerical values that these variables have, and they will indicate what the best storage policy is. The Performance variables vary from variables concerning the structure of the warehouse to the total energy consumption calculated during the simulation. The most relevant performance variables for this study are listed below.

- Throughput [orders/h]

  It is defined as the number of elements that enter or exit the system in a given time unit (Colla V. and Nastasi G., 2010). In the present study the throughput indicates the number of orders that the system is able to fulfil in the time unit, that is the number of orders that exit the system in the time unit (hours).

- Receptivity [units]

  It is the maximum number of items that can be stored in the warehouse (Colla V. and Nastasi G., 2010).

- Selectivity [%]

  It is a percentage equal to the directly reachable unit loads divided by the total unit loads stored in the warehouse.

- Shelf Occupation [%]

  It is a percentage that indicates the space occupied by the items, it is equal to the space occupied by items divided by the total space available.

- Unoccupied space [%]

  It is a percentage that indicates the unoccupied space, it is equal to the unoccupied space divided by the total space available.

- Vehicle utilization [%]

In the model this index has been calculated as the total activity time of the vehicle including travel time, picking time divided by the total time included vehicle activity time and vehicle idle time.

- Average Order Cycle time [min/order]

Cycle time is the time between the arrival of an order request to the time the request has been fully satisfied and fulfilled (Ekren B. Y. and Heragu, 2011). In the model the cycle time is the time between the order arrival and the time the mixed pallet is ready and unloaded in the buffer out, it includes all the waiting times that might happen during the picking activity.

- Average order Task time (Picking) [min/order]

It has been calculated as the time it takes to complete the picking activities without considering the waiting times divided by the total number of picking orders fulfilled.

- Average order Task time (Retrieval) [min/order]

It has been calculated as the time it takes to complete the retrieval activities without considering the waiting times divided by the total number of retrieval orders fulfilled.

- Average order waiting time [min/order]

It has been calculated as the total waiting time of fulfilled orders divided by the number of fulfilled orders.

- Directly reachable pallets [units]

Number of pallets that can be directly reached by the vehicle. It is useful to calculate the index of selectivity.

- N pallets stored [unit]

Total number of stored pallets in the warehouse. It is useful to calculate the index of occupied/unoccupied space

- Area Occupation [m3]

It indicates the volume in m^3 occupied by the warehouse

- Area Occupation [m2]

It indicates the area in m^2 occupied by the warehouse

- Average meters run by vehicles [m/n°vehicles]

It has been calculated as the total number of meters run by the vehicles divided by the number of vehicles.

- Average Energy consumption per vehicle [KWh/n°vehicle]

The calculation of the Average energy consumption has been carried out following some studies conducted on the same topic. Considering the paper written by Akpunar A. et al. the necessary KWh in an AVS/RS system during travel with constant velocity can be calculated as $Wc = p_c \times t$. Pc is the power needed to overcome the traction force in travel with constant velocity and t is equal to the time the vehicle has travelled with constant velocity (Akpunar A. et al., 2017). The paper of Bruno G. et al. gives an indication of the power needed by the AVS/RS shuttle when the shuttle is empty (1 KW) and when it is loaded (2KW) (Bruno G. et al., 2016). These elements have been useful to calculate the energy consumption in the simulation model. Assuming that the maximum load that can be put on a pallet is equal to 1500 kg and assuming that the power at load in the article of Bruno G et al. is related to full load, the power necessary to move 1500kg would be equal to $p_e - p_f$ where pe is the power when the shuttle is empty, and pf is the power needed when the shuttle is fully loaded. This means that every kg requires a power of $\frac{pe-pf}{1500}$. Knowing this information and knowing the time travels of the vehicles in the model, it is possible to calculate the energy consumption of the vehicles during the simulation, taking into account the different weights that the vehicles carry during the picking activity. The index of average energy consumption also includes an indication of the vehicles' passive energy, which is the energy used when the vehicles are on but they are not performing any activity.

5.3 Structure of Simulation model

To better understand how the simulation model works, it is important to describe its structure. Before running any simulation, the right model parameters are set so that through a code, triggered at simulation start, the model knows how many corridors, bays, pallets in each bay, levels, vehicles to build in the 3D model. It is also important to tell the model what storage policy it has to replicate. All this information is set in a specific table called “Model parameters Table”, showed in Figure 16.

Parameters 12

Name	Value
Storage_policy	1
Frequenza_ordini_misti	100
Frequenza_ordini_interi	300
N_sku	5
Numero_baie	15
Numero_livelli	3
Numero_slot	1
Numero_corridoi	2
Numero_Agv	3
Distanza_tra_parcheggi	3
Numero_pallet_slot	10
Aggiorno_tabella	5

Figure 16: Model Parameters table

As it is possible to observe, the model parameters in this case are: the storage policy (1 for Random, 2 for Class based storage, 3 for Dedicated slot storage, 4 for storage by weight and 5 for storage by association rules), the inter-arrival time for picking and retrieval orders, number of SKUs, physical structure of the racks. Some of these parameters will be modified to test different scenarios as already explained in the previous sections.

Having this information as input, the software is able to build the 3D model and together with the elementary elements of the warehouse, the initial code will also create the path that the vehicles can follow during the simulation and it will also create the so called “control points”. Control points are areas in which the vehicle makes some sort of decision, whether it is about loading/unloading of items/pallets or waiting to proceed with the next activity. For instance, they can be spotted in front of every slot address. In the model, the control points are extremely important because they have been used as a way to calculate the total amount of meters that the vehicles have run during the simulation. Once the 3D

structure is in place, the Process Flow animates the simulation giving an order and a sequence to the different activities to perform. Based on the storage policy specified in the Model parameters Table, the software activates specific process flows through a conditional decision. A simplified view of this approach is showed in Figure 17.

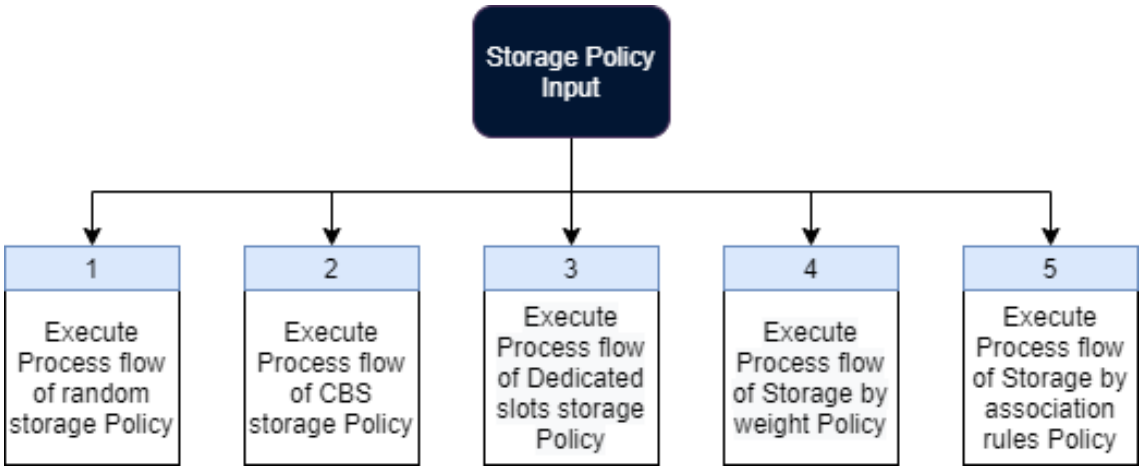


Figure 17: Process Flow approach

The raw model process flow is similar across all five storage policies tested in this study. What differentiates the five storage policies lies in the details of the code that has been written in FlexScript, a simplified coding language used in FlexSim, and in C++ and inserted in specific parts of the Process Flow. In the next sections there will be a general description of the Process Flow, which had already been built before the present study, followed by details on the custom codes that enable the simulation and comparison of different storage policies, specifically created for this thesis. The basic structure of the process flow is divided into two big areas: Initialization of data which include the actions of automatically populating the warehouse and the picking area which include orders generation, material handling and conclusion of picking order.

5.3.1 Initializing data

The process flow is initiated with the creation of a token which is an indication of the progress of the activity of warehouse filling, the token creation is not visible in Figure 18, but it is right before the first custom code called “Assegnazione SKU casuale”. Figure 18 shows the process flow concerning the warehouse filling operations.

In this section of the process flow, every rack slot will receive a label that contains the information of the SKU that it will host because at this stage, the model is preparing all the necessary to fill the warehouse completely. The lists which contain the Shuttles, the Satellites and the Robots are created, and some preliminary analyses are conducted to create tables that will host information needed to calculate the model performance variables. Once these first preliminary actions are done, the token moves into the other process box called “Creation of initial pallets” (“Creazione pallet iniziali”). In this section for every address of the slot, there is the creation of a pallet which is then automatically loaded with 12 boxes. The label of the slot address, containing the SKU value, is passed to the pallet and then to the boxes, so that the pallet is assigned to a slot with the same SKU. The process continues until the warehouse has been filled, which means that all levels, all sides and all bays of the warehouse have been correctly filled with pallets. After that, the token is destroyed.

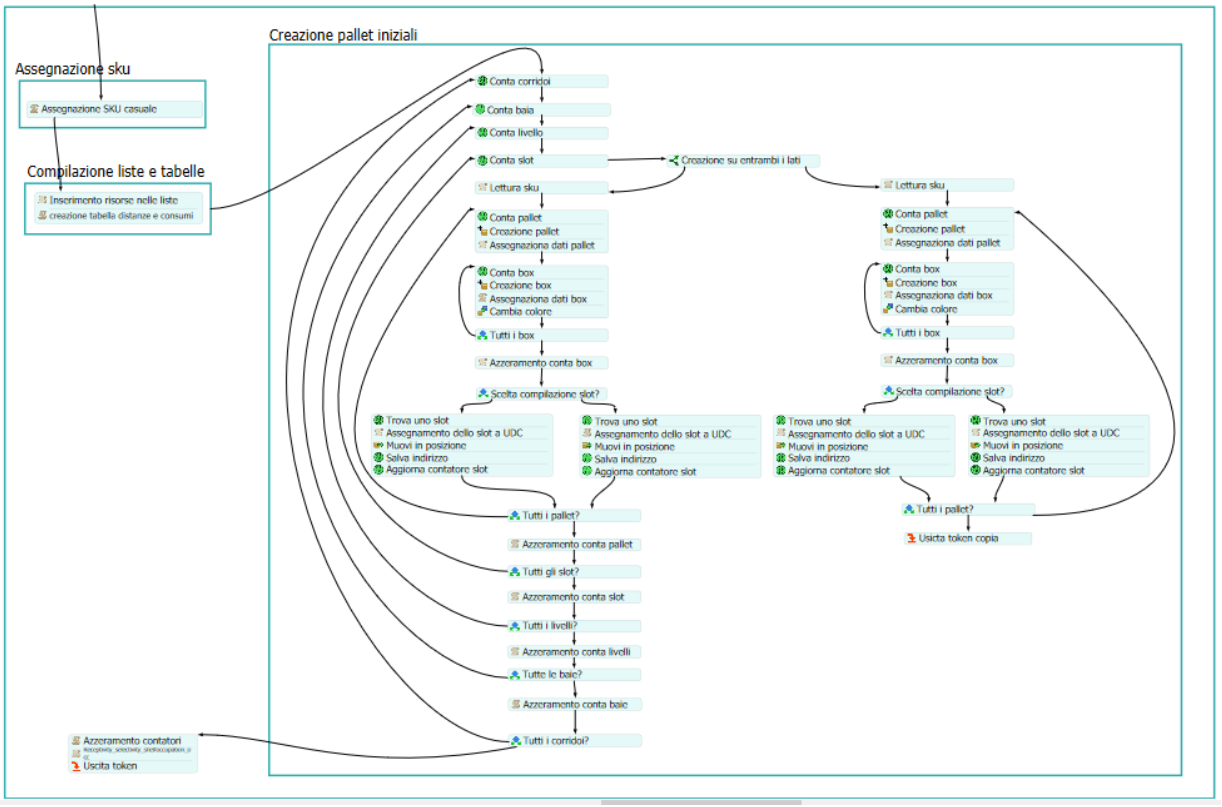


Figure 18: Process Flow Data initialization

5.3.2 Picking order Activities

Together with the creation of the token that enables the operation of filling the warehouse with pallets, there is the creation of tokens, with pre-specified interarrival token time, that trigger the creation of orders (picking orders and retrieval orders). As soon as the order is created, the activity of order picking starts. Figure 19 shows a fraction of the Process flow that handles the order fulfilment operations.

As already anticipated, the orders are created, and they are created in an Array format and saved in a table. The model works with the assumption that for every SKU, the client can order no more than three units (in this case three boxes). For instance, if the total number of SKU is five and the customer picking order is equal to [0,2,2,1,3] it means that the client is requesting 0 items of SKU1, 2 items of SKU2, 2 items of SKU3, 1 item of SKU4 and 3 items of SKU5. The client could not request more than 3 items per SKU.

The modifications of the model carried out specifically for this study include a very specific way of picking order generation. The model takes as input some past orders, it analyses the historical data by calculating the probability of SKU appearance in an order and it generates new orders based on that probability following a discrete empirical distribution. In this way the new order generated are not randomly created but they follow the behaviour of past data.

For every order generated, a pallet is created in a specific buffer and a specific resource (in this case a shuttle with a satellite and a robotic arm) is acquired and assigned to that specific customer order. The model reads the generated order and some labels, which store information on the SKU to pick and in which quantity, are created and assigned to the token. As soon as the model understand what the first SKU to pick is and how many boxes to pick, the handling operations can start. The handling operations vary a bit if the vehicle is dealing with the first SKU picking activity or if the vehicle is in the middle of satisfying an order. When the vehicle has completed the picking for the first SKU, the token goes back to the “Gestione cicli picking” box in order to read the next SKU to pick and its quantity for that same order. Once all the SKUs have been read and all the handling operations have been completed, the vehicle finds itself with a mixed pallet that needs to be discharged in a specific buffer for completed orders. Once this is done, the vehicle travels back to its

parking spot, it is released, and it waits for a new order to fulfil and some important information on the operations that just happened are saved in dedicated tables.

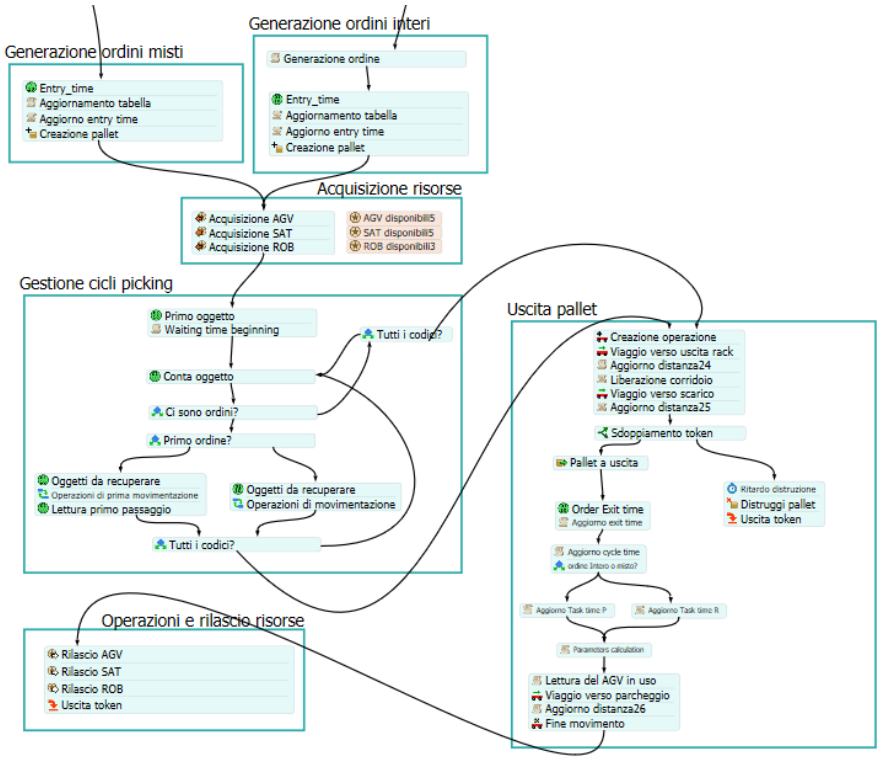


Figure 19: Process Flow Order picking

5.3.3 Handling Operations

The handling operations are triggered in the “Gestione cicli picking” box whenever the customer order requires boxes of a specific SKU to be picked. Figure 20 aims at showing the complexity of the handling operations, they will not be described in detail, but they are certainly an important part that enables the vehicles to move around the model by reaching prebuilt control points. The information boxes that include the handling operations are mainly instructions given to the vehicle for it to reach the correct slot address with the right SKU to pick. It is a process that triggers travel activities, loading activities and picking activities. As soon as the model identifies the right slot for the vehicle, the vehicle travels and it reaches the destination. The loading activity is created in the first iteration of the handling process when the vehicle loads on itself the pallet onto which to put the boxes.

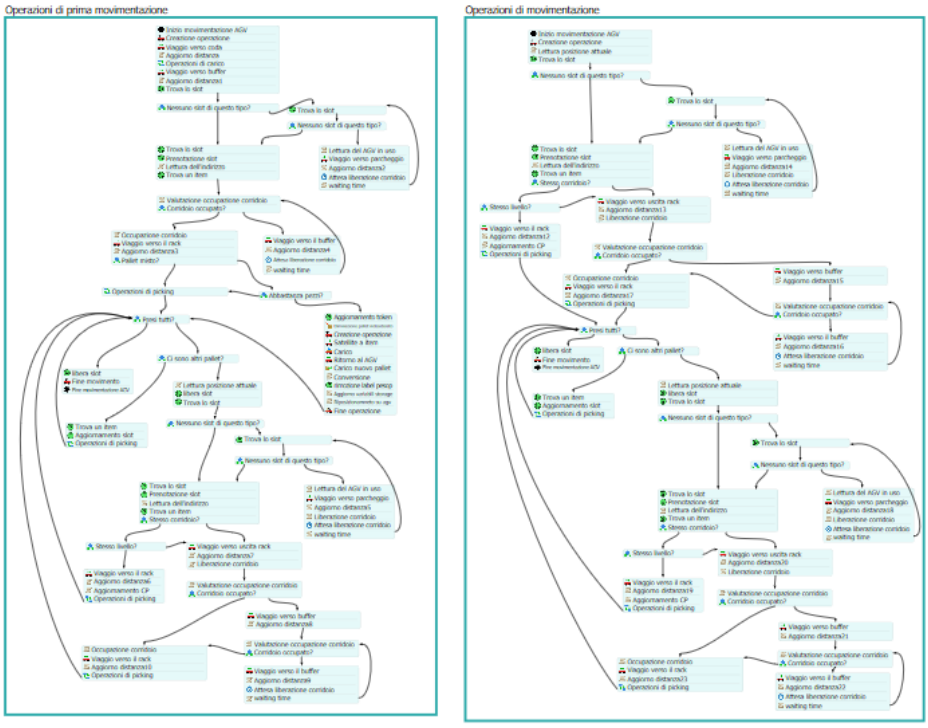


Figure 20: Process Flow First Handling operation and Handling Operation

The picking activity is activated when the vehicle has reached the correct slot address and the satellite needs to bring the pallet stored in that address to the shuttle and the robotic arm needs to pick the needed boxes from the loaded pallet. During the handling process flow, some custom codes have been added so that important information on the behaviour of the vehicles, such as meters run and energy consumption, can be accurately stored in specific tables.

Figure 21 shows the picking operation, which is one of the most important activity that is triggered during the handling operations process flow. As soon as the vehicle has reached the slot hosting the unit load with the correct SKU to pick, the operation starts. The satellite detaches from the vehicle, it loads the stored pallet, and it goes back to the vehicle. The model verifies whether there are enough boxes in the stored unit load. If yes, the robotic arm picks the boxes from the unit load and it place them on the pallet resting on the vehicle. After that, the satellite unloads the unit load back to its slot. If there are not enough boxes on the stored unit load, the robotic arm picks all the boxes that the unit load can offer, and the token will trigger the handling operations in order to find another stored unit load with the same SKU. In this way it is possible to pick the remaining number of boxes for that SKU and satisfy the request in the order.

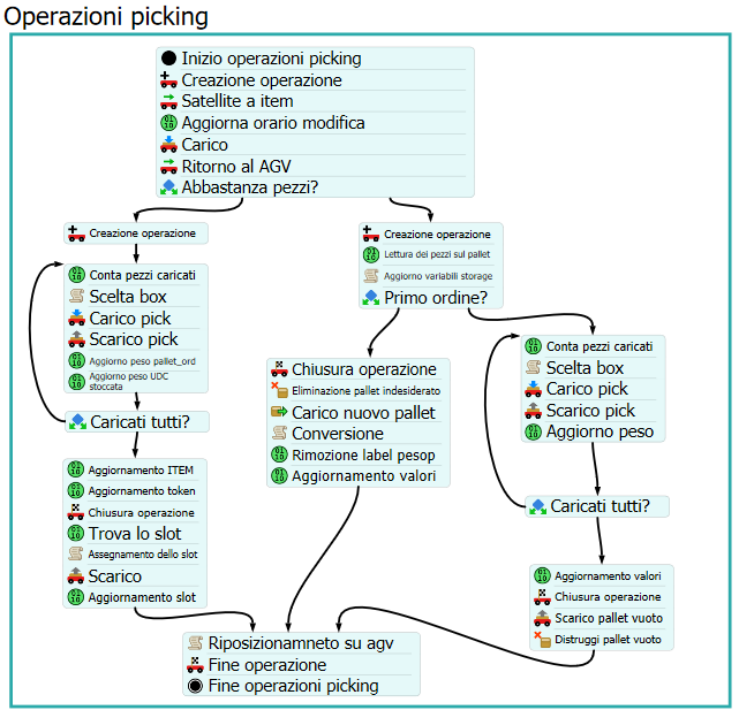


Figure 21: Process Flow Picking operation

5.3.4 Custom codes in Random Storage Policy

In the Random Storage Policy every slot is filled with ten unit loads and every unit load is characterized by a random SKU. In this storage policy, the codes give a random SKU value to the unit loads and the SKU then is associated to the slot. Since the total number of unit loads is 10, the slot has in total 10 labels called SKU, SKU1, SKU2, SKU3 until SKU9. Every time one unit load is created in the slot, its SKU is saved. When the first unit load is inserted (considered as the 10^{th} in line in the slot), SKU1 will be equal to the SKU of that unit load. The same works for all the other unit loads so that for instance, when the last unit load of the slot is inserted (the most accessible unit load and the one closer to the corridor), the label SKU of the slot will be set equal to the SKU of that unit load. The vehicle, when looking for a specific SKU, will look for the right SKU by reading the slots labels. The following code shows how the SKU is given to the slot when the first unit load is created in the slot, a very similar code is applied for every unit load but, it will not be shown to avoid unnecessary repetitions.

```
if(token.cpal==1)
{
 token.UDC.num = 1;
 if (token.cbai<=9)
```

```
{
 caddressA = concat(string.fromNum(token.ccor),"-1-
0",string.fromNum(token.cbai),"-",string.fromNum(token.cliv),"-
",string.fromNum(token.cslo));
 Storage.system.getSlot(caddressA).SKU1 = token.UDC.sku;
 }
 else
 {
 caddressA = concat(string.fromNum(token.ccor),"-1-
",string.fromNum(token.cbai),"-",string.fromNum(token.cliv),"-
",string.fromNum(token.cslo));
 Storage.system.getSlot(caddressA).SKU1 = token.UDC.sku;
 }
}
```

Whenever a unit load is empty, the SKU of the slot will be set equal to the next unit load SKU so that the correct SKU picking is always guaranteed. As an example, if one unit load has been fully emptied and the slot contains nine full pallets and not ten, the SKU of the slot should not be equal to the empty unit load anymore, but it should be equal the one next in line. The code counts for all the unit loads in the slot and when it realizes that there are nine unit loads, instead of ten, the SKU of the slot is set equal to the 9th unit slot SKU, SKU9. If the counted unit loads of the slot are equal to eight, then the SKU of the slot should be set equal to the SKU of the 8th unit load, SKU8. The same reasoning applies when the unit loads in the slot are lower than eight. The following code shows how the right SKU is applied to the slot whenever a unit load is emptied, where “ind” indicates the address of the slot and “oggetti_presenti” are the number of unit loads in the slot.

```
if (oggetti_presenti==9)
{
Storage.system.getSlot(ind).SKU =Storage.system.getSlot(ind).SKU9;
}
```

Figure 22 shows what the Random Storage Policy looks like in the FlexSim Simulation Model when the warehouse has one corridor, three levels, three vehicles, 15 bays and five SKUs (differentiated by colour).

In this Storage Policy the vehicles satisfy the orders starting from SKU1 and ending with SKU5 if there is a total of five SKUs. No particular prioritization instructions have been given to the vehicles when dealing with picking order activities.

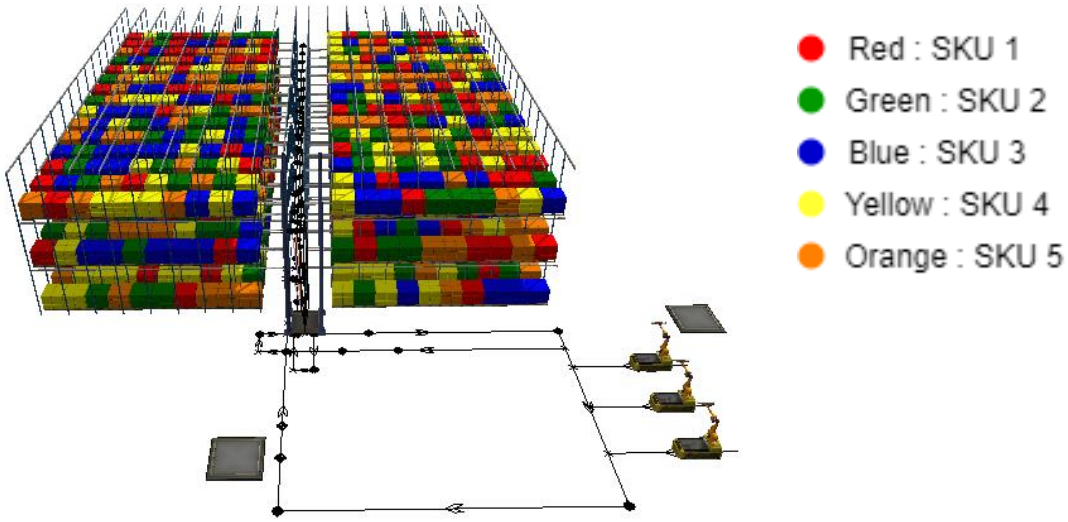


Figure 22: Implementation of Random Storage Policy

5.3.5 Custom codes in Class Based Storage Policy

In the Class Based Storage Policy, the model deeply analyses past data fed as input in order to assign each SKU to the right area. The code specifically written for this storage policy counts all the occurrences of the SKUs in the past orders, it orders the findings from highest output to lowest output and a cumulative share is calculated. All the SKUs that contribute to a cumulative share lower than 60% are part of class A, the ones who contribute to a cumulative share which is between 60% and 90% are part of class B and lastly the SKUs which contribute to a cumulative share between 90% and 100% are part of class C. Knowing this information, the model stores the SKUs belonging to class A, closer to the buffer out, it stores the SKUs of class B right after the ones in class A and in the end the SKUs from class C are taken into account. In order to follow this storage logic, it has been necessary to save all the distances between the control points of the slots and the buffer out, order them in ascending order and gradually assign them to the SKUs in class A, then class B, and then class C. If more than one SKUs were belonging to the same class, the slot SKU from that class was assigned randomly between the available SKUs. Figure 23 shows what the Class Based storage looks like in the model. The slots closer to the buffer out are dedicated to the SKUs belonging to class A (in this case SKU 1 and SKU 3), these two SKUs have been distributed randomly in the Area A. Class B is dedicated to SKU 4 and SKU2 and the last

class, class C hosts SKU 5. The division of the SKUs into different classes will vary if the past orders, fed as input into the model, change.

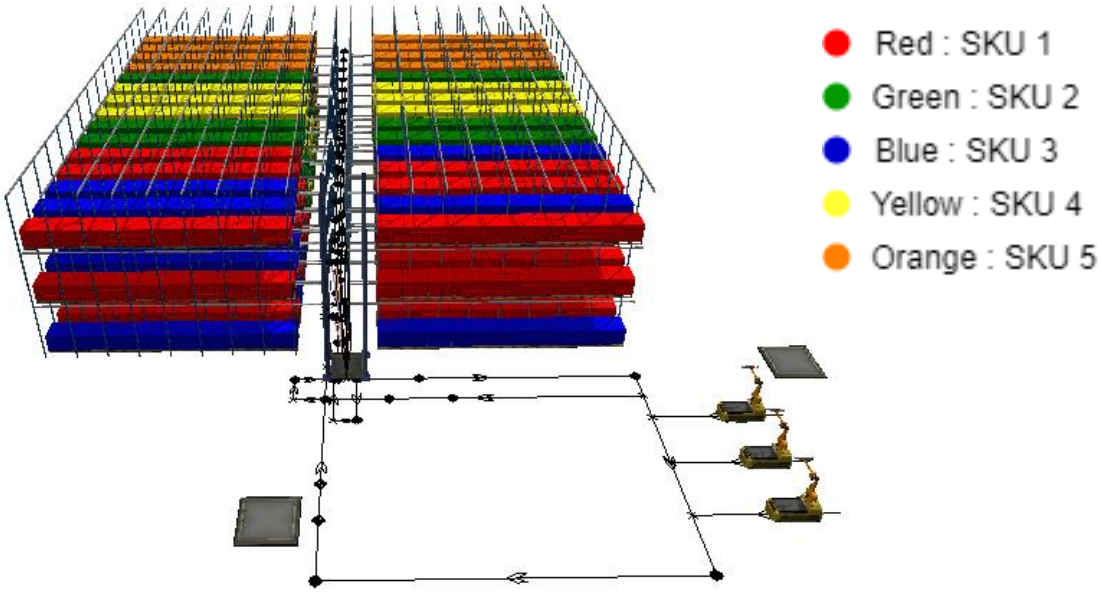


Figure 23: Implementation of Class Based Storage Policy

The following code partially shows how the slot assignment has been performed for class A. A very similar approach has been used for class B and C. The code reads the SKUs belonging to the class A after having analysed the past orders. It calculates how many slots it has to save for class A considering the total number of SKUs belonging to class A. It then reads a table which stores all the distances from the control points to the buffer out in ascending order and it randomly assign the slots in class A to the SKUs belonging to class A using a discrete uniform distribution.

```
for (int i=1; i<= Table("Tabella_CP").numRows; i++)
{
      /*class A*/
      if(counter1 < num_pallet_sku*a)
      {
      indirizzo = Table("Tabella_CP")[i][1];
      indici = indirizzo.split();
```

```
      caddressA = concat(indici[3],"-1-",indici[5],indici[6],"-
",indici[4],"-",indici[7]);
      posizione = duniform(1,a);
      skudaassegnare = A[posizione];
      Storage.system.getSlot(caddressA).SKU = skudaassegnare;
      caddressB = concat(indici[3],"-2-",indici[5],indici[6],"-
",indici[4],"-",indici[7]);
      posizione = duniform(1,a);
      skudaassegnare = A[posizione];
      Storage.system.getSlot(caddressB).SKU = skudaassegnare;


      counter1 = counter1 + Model.parameters.Numero_pallet_slot*2;
      l=i;
      }

}
```

It is important to underline that in this storage policy, the vehicles, when satisfying a picking order, they give priority to the SKUs that belong to class A, then they deal with the ones in class B and lastly the care about the ones in class C. This allows to further optimize the path that the vehicles follow during the picking order activity and it allows to maximize the benefits of this storage policy.

5.3.6 Custom codes in Dedicated slots Storage Policy

In this storage policy which was part of the original version of the model, the label with the SKU information is assigned randomly to the slots and, contrarily to what happens in the random storage policy, the slots are filled with unit loads that have the same SKU. The code below partially shows how this assignment is performed and Figure 24 shows what the Dedicated Slots storage looks like in the model.

```
caddressA =
concat(string.fromNum(ccor),"10",string.fromNum(cbay),"-
",string.fromNum(clevel),"-",string.fromNum(cslot));

skudaassegnare = duniform(1,Model.parameters.N_sku);
Storage.system.getSlot(caddressA).SKU = skudaassegnare;

caddressB =
concat(string.fromNum(ccor),"-2-0",string.fromNum(cbay),"-
",string.fromNum(clevel),"-",string.fromNum(cslot));

skudaassegnare = duniform(1,Model.parameters.N_sku);
Storage.system.getSlot(caddressB).SKU = skudaassegnare;
```

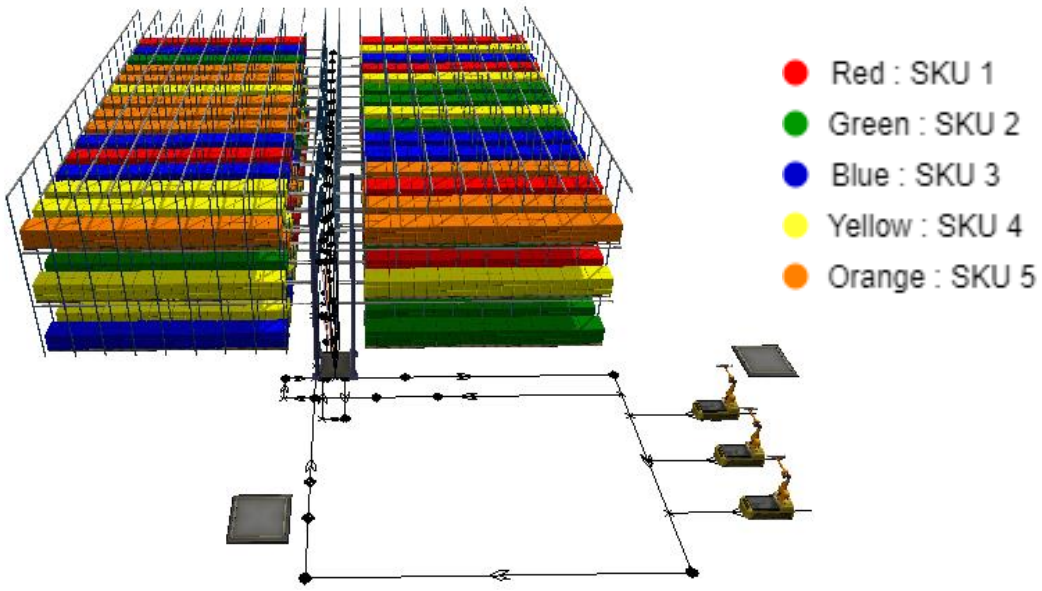


Figure 24: Implementation of Dedicated slots storage

In this Storage Policy the vehicles satisfy the orders starting from SKU1 and ending with SKU5 if there is a total of five SKUs. No prioritization instructions have been given to the vehicles when dealing with picking order activities.

5.3.7 Custom codes in Storage by weight Policy

For this storage policy, the weight of the single boxes is extremely important because the weight determines where the SKU will be stored. In the simulation model, boxes from SKU 4 (SKU yellow) are the heaviest ones and they are stored closest to the buffer out, the lightest SKUs will be stored far away from the buffer out. The code below shows how the unit loads have been stored. The code reads what the heaviest SKU is, and it goes through a table that contains the distances between the control points and the buffer out and it assigns the slots with the shortest distance to the heaviest SKUs.

```
for ( int k=1; k <= Model.parameters.N_sku; k++)
{
    j = 0;

      for (int i=1; i<= Table("Tabella_CP").numRows; i++)
      {
            if(j<num_pallet_sku)
            {
             indirizzo = Table("Tabella_CP")[i][1];
```

```
                indici = indirizzo.split();

            caddressA = concat(indici[3],"-1-
",indici[5],indici[6],"-",indici[4],"-",indici[7]);
            skudaassegnare = weights[k][1];
            Storage.system.getSlot(caddressA).SKU = skudaassegnare;
            caddressB = concat(indici[3],"-2-
",indici[5],indici[6],"-",indici[4],"-",indici[7]);
            skudaassegnare = weights[k][1];
            Storage.system.getSlot(caddressB).SKU = skudaassegnare;

            l=l+1;
            j = j + Model.parameters.Numero_pallet_slot*2;

             }
      }
}
```

Figure 25 shows what the Storage by weight looks like considering that in the model, SKU 4 weights 50 kg, SKU 3 weights 40 kg, SKU 5 and 2 weight 30 kg and SKU 1 weights 20kg.

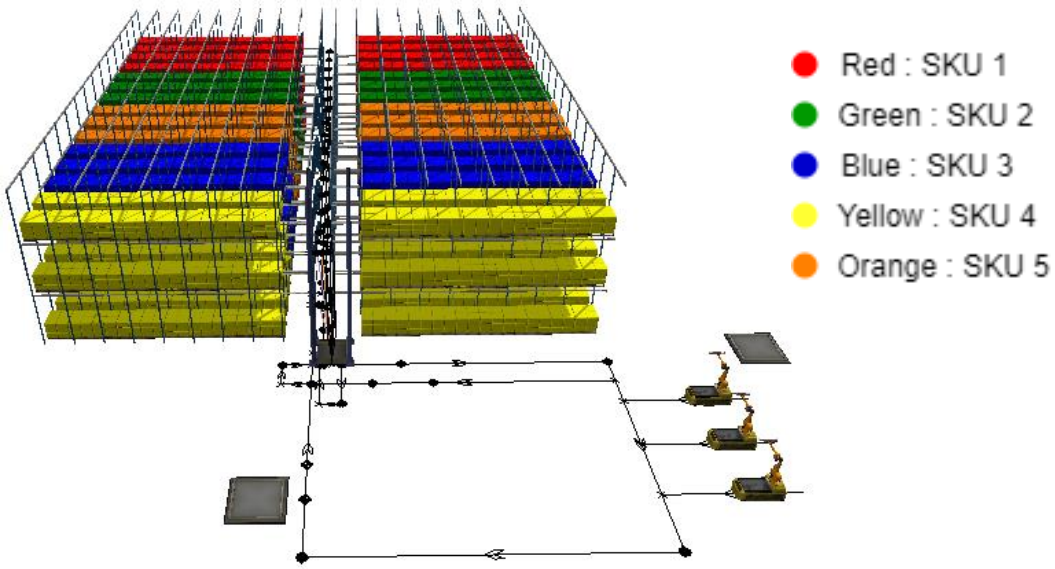


Figure 25: Implementation of Dedicated slots storage

It is probably straight forward to understand the benefits of this storage policy when only dealing with retrieval activities. Since in this study the focus is mainly on picking order activities, the vehicles follow some prioritization rules when fulfilling an order. In this way it is possible to maximize the benefits of this storage policy when dealing with picking orders. The vehicle will read how many boxes for each SKUs are required for that specific order and it will calculate the total weight of the SKUs, which depends on the quantity of

boxes required and on the weight of a single box. It will then start picking the SKUs from the ones that overall weight less. This implies that the sequence followed by the vehicle is linked to the order that it is fulfilling and therefore it will most likely change every time that the vehicle deals with a new order.

5.3.8 Custom codes in Storage by association rules Policy

In this storage policy the model analyses past order and on them it applies the *Apriori* algorithm in order to identify the combination of SKUs that happen most frequently in the customer orders. Having this information, the model stores the SKUs with highest frequency close to each other. In order to maximize the benefits of this storage policy, the vehicles will give prioritization to the SKUs identified as the ones that most frequently appear together in a customer order and it will then pick the remaining SKUs. The code below is a simplified version of what has been written in the model, but it still gives an idea on the approach followed to store the unit loads in this storage policy. The code saves in one array the SKUs that have been identified through the *Apriori* algorithm and it will put in another array the remaining SKUs. The code will then fill the slots one by one by always storing the SKUs in the separate arrays together.

```
for(int i=1;i<=v.length;i++)
{
      caddressA = concat(string.fromNum(ccor),"-1-
0",string.fromNum(cbay),"-",string.fromNum(clevel),"-
",string.fromNum(cslot));

      caddressB = concat(string.fromNum(ccor),"-2-
0",string.fromNum(cbay),"-",string.fromNum(clevel),"-
",string.fromNum(cslot));

      skudaassegnare= stringtonum(v[i]);
      Storage.system.getSlot(caddressA).SKU = skudaassegnare;
      Storage.system.getSlot(caddressB).SKU = skudaassegnare;
      cbay=cbay+1;


for(int i=1;i<=k.length;i++)
{

       caddressA = concat(string.fromNum(ccor),"-1-
",string.fromNum(cbay),"-",string.fromNum(clevel),"-
",string.fromNum(cslot));
```

```
        caddressB = concat(string.fromNum(ccor),"-2-
",string.fromNum(cbay),"-",string.fromNum(clevel),"-
",string.fromNum(cslot));

      skudaassegnare=k[i];
      Storage.system.getSlot(caddressA).SKU = skudaassegnare;
      Storage.system.getSlot(caddressB).SKU = skudaassegnare;
      cbay=cbay+1;

}
```

Figure 26 shows what the storage by association rules look like in the model, keeping in mind that through the past orders analysis, it has been discovered that the SKUs that appear most frequently together in an order are SKU1 and SKU3.

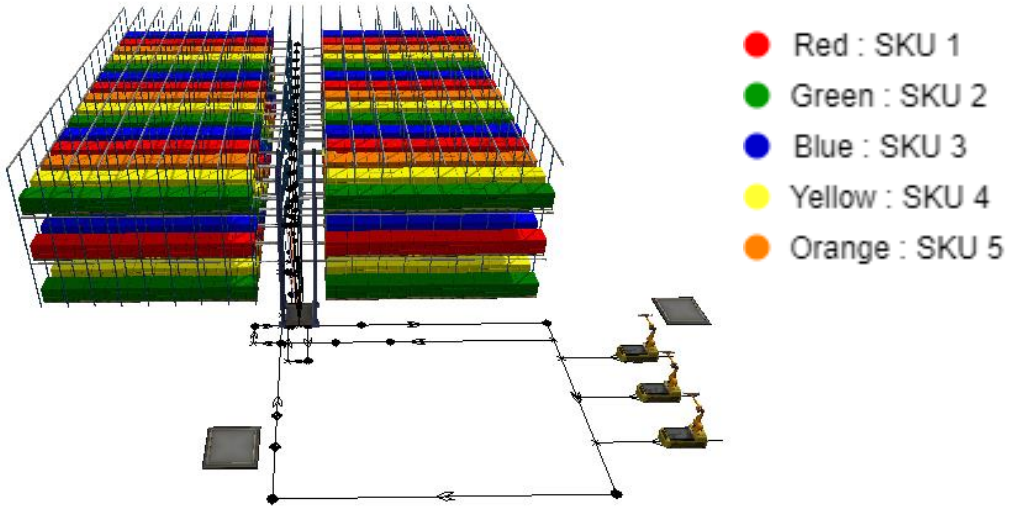


Figure 26: Implementation of Storage by association rules

5.3.9 Custom codes for Performance variables calculation

The original model has been modified not only in the way unit loads are stored and in the way they are chosen by the vehicles, but important modifications have been introduced in order to calculate in real time, relevant performance variables.

As soon as the process flow of warehouse filling ends all its loops, a custom code calculates the most important warehouse characteristics like receptivity, selectivity, occupied/unoccupied shelves. These indexes are regularly updated as the simulation proceeds so that they correctly mirror the warehouse changes.

The vehicles utilization is calculated by tracking the state of each vehicle through a function in FlexScript as showed in the code below. The utilization is given by the time used by the vehicle to perform its activities such as travel, loading, picking, divided by the total time of the vehicle including idle time. In this way it is possible to have a fraction of work time over total vehicle time.

```
a = Model.find(s).as(Object).stats.state().getTotalTimeAt(STATE_ALLOCATED_IDLE);
I = Model.find(s).as(Object).stats.state().getTotalTimeAt(STATE_IDLE);
b = Model.find(s).as(Object).stats.state().getTotalTimeAt(STATE_BLOCKED);
t = Model.find(s).as(Object).stats.state().getTotalTimeAt(STATE_TRAVEL_LOADED);
```

For the vehicle meters and vehicle consumption it was necessary to create some dedicated tables in order to store the relevant information in the clearest way. No matter what kind of storage policy will be used, the model will create a Table aimed at containing the meters run by each vehicle and a Table containing the energy consumptions in KWs. These data will be used in the end to calculate the energy consumption in KWh. The Tables are organized in a way that show how much energy and how many meters have been consumed and run by the vehicles when carrying different weights during the order fulfilment. An example of these data organization is showed in Table 4 and Table 5.

Table 4: Meters run by all vehicles with different loads

	Empty	box1	box2	box3	box4	box5	box6	box7	box8	box9	box10	box11	box12	box13	box14	box15
AGV1	563.9073	16	14	8	2	48.5	77.8246	0	0	0	0	0	0	0	0	0
AGV2	579.9868	0	18	8	26.5	2	30.6623	13.1623	0	0	0	0	47.1623	0	0	0
AGV3	534.7483	0	4	29.5828	8	6	2	0	27.5828	27.5828	0	0	20.5	0	0	0

Table 5: Energy consumed by all vehicles with different loads

	Empty	box1	box2	box3	box4	box5	box6	box7	box8	box9	box10	box11	box12	box13	box14	box15
AGV1	563.9073	16.2133	14.3733	8.9867	2.32	61.8133	99.3822	0	0	0	0	0	0	0	0	0
AGV2	579.9868	0	20.1867	8.32	33.3667	2.48	39.7454	16.1457	0	0	0	0	71.6682	0	0	0
AGV3	534.7483	0	4.1067	32.4716	8.8533	7.04	2.2933	0	36.2548	40.8225	0	0	40.18	0	0	0

Every time a vehicle travel activity is concluded so, every time the vehicle reaches a new control point, the values in the tables are updated.

In the Process Flow there are other custom code that keep track of when the orders first appear in the simulation as requests and when the orders are completed and the physical mixed or retrieved pallet is discharged. This information is stored in the Table containing the list of incoming orders and it helps in calculating the Cycle time and the Task time of order fulfilling operations. Whenever an order is waiting to be processed or whenever a vehicle with an order is waiting for a specific corridor to be freed, a code saves the start

and end time of that wait so that it is possible to quantify the waiting time of each request. Table 6 shows how the order information is organized in the Table, all the data are then used to calculate the average order cycle time, average order task time and average order waiting time.

Table 6: Information about incoming orders

Orders	Entry_time	Exit_time	Waiting time at the beginning	Waiting time during picking	Cycle time	Task time Picking	Task time Retrieval
Array[5]: {2, 1, 2, 1, 0}	1	323.0587	0	0	322.0587	322.0587	0
Array[5]: {0, 0, 12, 0, 0}	51	221.5662	0	0	170.5662	0	170.5662
Array[5]: {3, 2, 1, 2, 0}	101	471.4933	0	0	370.4933	370.4933	0
Array[5]: {0, 0, 2, 2, 0}	201	496.9933	31.5662	0	295.9933	264.4271	0
Array[5]: {1, 2, 0, 2, 1}	301	622.7169	30.0587	0	321.7169	291.6582	0
Array[5]: {0, 0, 0, 12, 0}	351	669.3154	134.4933	0	318.3154	0	183.8221

6. Execution of Simulation model and discussion of results

All the simulations performed in the study have given different outputs that will be deeply analysed in this section. The focus will mostly be set on the behaviour of Throughput, Cycle time, Vehicle utilization and energy consumption under two macro scenarios: scenario with retrieval and picking orders and scenario with picking orders only. In both scenarios there will be the analysis of the outputs of the storage systems taken independently from each other. In this first part, the study wants to understand how the single storage policies behave when the input parameters are varied, if the output trend is the same for all storage policies, then, only one storage policy will be deeply analysed. After that, there will be an output comparison between all five different storage systems to determine the best performing one.

6.1 Scenario 1: Retrieval and Picking orders

These analyses have been performed by considering that both picking orders and retrieval orders enter the warehouse at specific and pre-determined interarrival times. As previously mentioned, the interarrival time is one of the input parameters that will be changed in different simulations and it will be set equal to 50 seconds in some cases and equal to 100 seconds in other ones. On the other hand, the interarrival time of retrieval orders is always set equal to 300 seconds.

As previously explained, for every storage policy, 32 simulations have been performed, each one of them having different input parameters. Every single simulation is an average of other five simulations in order to get results as reliable as possible, an example is given by Table 7. For every storage policy a Table like Table 8 has been created and it gathers all results from the 32 simulations. For simplicity only one storage policy will be analysed independently, in this case the Random policy, to understand its behaviour with different input parameters. If, however, the other storage policies show different patterns compared to the Random storage policy, this difference will be highlighted.

Table 7: An example of a simulation given by the average of other five simulations

Simulation 1						
Throughput [orders/h]	32.2304	31.902	32.554	31.7395	33.2462	32.33442
Receptivity [units]	1800	1800	1800	1800	1800	1800
Selectivity [%]	0.0978	0.0978	0.0978	0.0978	0.0989	0.09802
Shelf Occupation [%]	0.9	0.9006	0.8978	0.9033	0.9033	0.901
Unoccupied space [%]	0.1	0.0994	0.1022	0.0967	0.0967	0.099
AGV utilization [%]	0.9738	0.9715	0.9759	0.9746	0.9729	0.97374
Avg Order Cycle time [min/order]	176.9087	183.7014	182.5969	178.5851	178.5179	180.062
Avg order Task time (Picking) [min/order]	5.5307	5.6584	5.4917	5.6523	5.3993	5.54648
Avg order Task time (Retrieval) [min/order]	4.0738	3.9815	4.2379	4.0536	4.0289	4.07514
Avg order waiting time [min/order]	172.7254	178.2664	177.2691	174.3416	173.3161	175.1837
Directly reachable pallets [units]	176	176	176	176	178	176.4
N pallets stored [unit]	1620	1621	1616	1626	1626	1621.8
Area Occupation [m3]	15660	15660	15660	15660	15660	15660
Area Occupation [m2]	2384	2384	2384	2384	2384	2384
Avg meters run by Agvs [m/Agv]	19589.1092	19754.57	18767.45	19472.93	19656.46	19448.11
Avg Energy consumption per Agv [KWh/Agv]	5.9318	5.9907	5.677	5.8514	5.9642	5.88302
Meters Agv1	19566.9312	19679.44	18438.98	19465.47	20137.41	19457.65
Meters Agv2	19723.0737	19557.58	18797.53	19581.75	19238.64	19379.71
Meters Agv3	19477.3227	20026.7	19065.84	19371.56	19593.34	19506.95
Energy consumption Agv1	5.9344	5.9267	5.5687	5.8377	6.0976	5.87302
Energy consumption Agv2	5.9609	5.9114	5.702	5.8898	5.84	5.86082
Energy consumption Agv3	5.9001	6.134	5.7603	5.8266	5.955	5.9152

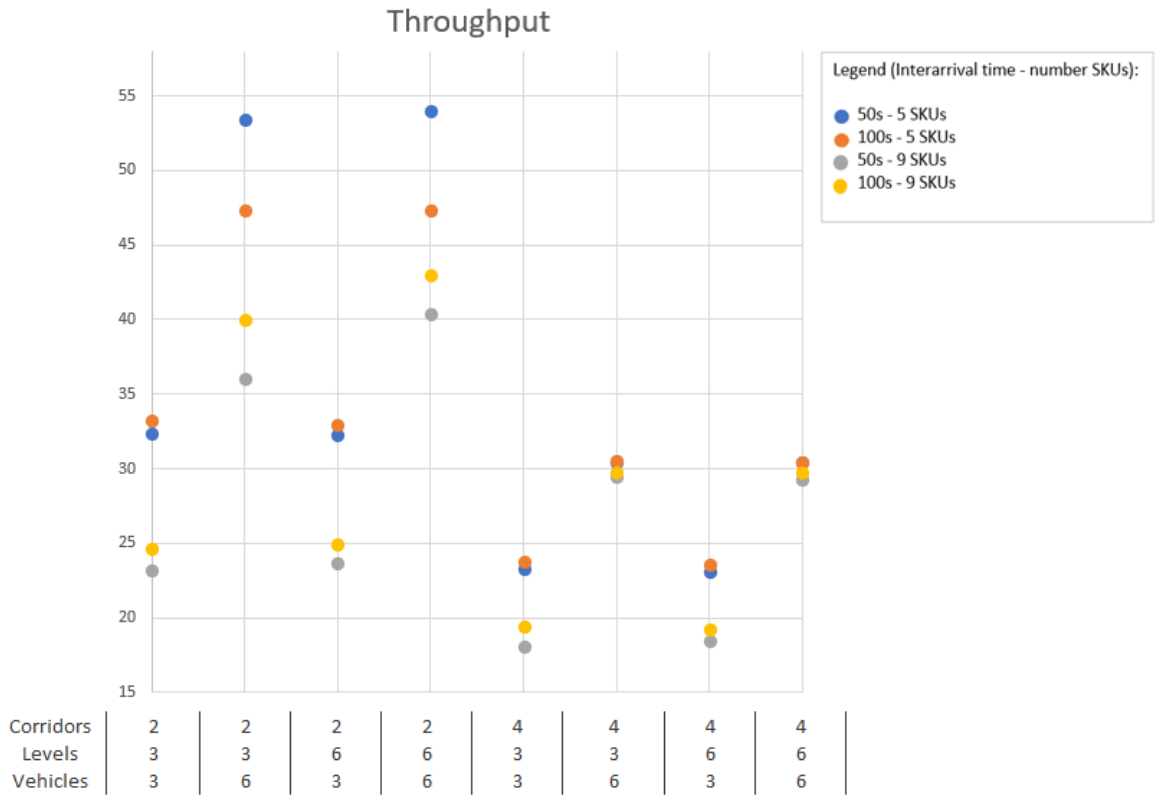
Table 8: Overview of 32 simulations for one storage policy

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
Throughput [orders/h]	32.33442	33.18594	23.17164	24.64066	53.36496	47.35552	36.0666	40.00584	32.23654	32.96626	23.71282	24.94796	54.00506	46.74654	40.36036	42.97092	23.25692	23.72948	18.10786	19.44574	30.47596	30.51352	29.4326	29.71652	23.13274	23.60254	18.50994	19.2743	30.45488	30.47944	29.29764	29.79766
Receptivity [units]	1800	1800	1800	1800	1800	1800	1800	1800	1800	1800	1800	1800	1800	1800	1800	1800	1800	1800	1800	1800	1800	1800	1800	1800	7200	7200	7200	7200	7200	7200	7200	7200
Selectivity [%]	0.09802	0.09612	0.09976	0.09964	0.09522	0.09502	0.09942	0.09756	0.09912	0.09854	0.09994	0.09982	0.0975	0.09786	0.09932	0.09862	0.09944	0.09928	0.1	0.09994	0.09928	0.0989	0.1	0.09964	0.09992	0.09966	0.1	0.1	0.09964	0.0994	0.09996	0.09972
Shelf Occupation [%]	0.901	0.88758	0.89756	0.88788	0.8311	0.837	0.83978	0.80944	0.95188	0.9455	0.9497	0.94502	0.91494	0.92502	0.9103	0.90596	0.96484	0.9619	0.96206	0.95866	0.95388	0.94936	0.93578	0.93506	0.98306	0.98126	0.98048	0.97974	0.977	0.975	0.96888	0.96664
Unoccupied space [%]	0.099	0.11242	0.10244	0.11212	0.1689	0.163	0.16022	0.19056	0.04812	0.0545	0.0503	0.05498	0.08506	0.07498	0.0897	0.09404	0.03516	0.0381	0.03794	0.04134	0.04612	0.05064	0.06422	0.06494	0.01694	0.01874	0.01952	0.02026	0.023	0.025	0.03112	0.03336
AGV utilization [%]	0.97374	0.97114	0.975	0.97694	0.9735	0.7602	0.9751	0.9653	0.9424	0.94084	0.9506	0.94516	0.9429	0.93194	0.94816	0.93464	0.98068	0.98174	0.98054	0.983	0.98528	0.98338	0.98326	0.98198	0.9594	0.95952	0.96484	0.96314	0.97288	0.97218	0.96646	0.96262
Avg Order Cycle time [min/order]	180.062	89.0973	209.971	138.2841	102.4319	5.79878	141.50766	39.54874	182.1038	92.35044	210.0304	138.346	105.72232	83.4583	153.3403	25.01408	216.2657	150.1638	236.3026	180.4924	196.04468	114.0643	197.9292	117.0223	218.3732	153.1207	233.3162	179.1967	195.4046	114.8315	198.1917	116.6948
Avg order Task time (Picking) [min/order]	5.54648	5.58472	8.00076	7.88074	6.5844	5.98984	9.7459	9.28578	5.59882	5.68416	7.89068	8.00346	6.5662	7.34726	9.03462	8.7369	7.7133	7.80556	10.20632	10.0178	11.60982	11.91878	12.32678	12.6796	7.76134	7.93826	10.01159	10.17152	11.67064	11.94634	12.42022	12.68908
Avg order Task time (Retrieval) [min/order]	4.07514	4.09504	3.9961	4.06424	4.81504	4.48276	4.70478	4.88094	4.06022	3.9036	4.03702	3.81756	4.9229	5.65004	4.74854	4.71332	6.29682	5.91646	6.1373	5.98374	10.1336	9.81766	8.23038	8.26616	6.099	5.63366	5.98434	5.71732	9.96434	9.75984	7.87438	8.2052
Avg order waiting time [min/order]	175.18372	84.00188	202.5441	131.3494	96.1695	0.17508	132.84206	31.89364	176.7235	87.09768	202.6932	131.858	99.62396	76.64066	145.2382	17.46226	208.7443	142.8244	227.2133	171.471	184.63994	102.6658	186.1533	106.3596	210.7436	145.7232	223.8711	170.1101	184.2242	103.5755	186.9474	105.2615
Directly reachable pallets [units]	176.4	173	179.6	179.4	171.4	171	179	175.6	356.8	354.8	350.8	359.4	351	352.2	357.6	355	358	357.4	360	359.8	357.4	356	360	358.8	719.4	717.6	720	720	717.4	715.6	719.6	718
N pallets stored [unit]	1621.8	1597.6	1615.6	1598.2	1496	1506.6	1511.6	1457	3426.8	3403.8	3419	3402	3293.8	3330	3277	3261.4	3473.4	3462.8	3463.4	3451.2	3434	3417.8	3368.8	3366.2	7078	7065	7059.4	7054.2	7034.4	7020	6976	6959.8
Area Occupation [m3]	15660	15660	15660	15660	15660	15660	15660	15660	31320	31320	31320	31320	31320	31320	31320	31320	31320	31320	31320	31320	31320	31320	31320	31320	62640	62640	62640	62640	62640	62640	62640	62640
Area Occupation [m2]	2384	2384	2384	2384	2780	2780	2780	2780	2384	2384	2384	2384	2780	2780	2780	2780	4936	4936	4936	4936	5854	5854	5854	5854	4936	4936	4936	4936	5854	5854	5854	5854
Avg meters run by Agvs [m/Agv]	19448.1055	19779.32	18206.52	18159.55	18665.71	16006.07	17392.059	18868.42	19780.48	20126.36	18083.63	19070.62	18635.415	19752.4	18016.82	18373.82	23185.26	23474.28	21333.58	21559.34	16268.583	15959.45	18339.84	18252.52	23375.14	23630.9	21190.88	21560.07	16063.39	15977.46	18738.6	18416.81
Avg Energy consumption per Agv [KWh/Agv]	5.88302	6.00082	5.7439	5.70544	5.65232	4.877	5.69214	5.93878	5.93794	6.08506	5.65892	5.73982	5.67185	4.8463	5.65024	5.71224	6.8069	6.9211	6.4697	6.49228	4.80856	4.72378	5.56626	5.5382	6.82454	6.91886	6.35948	6.44108	4.6941	4.65788	5.64806	5.54352
Meters Agv1	19457.649	19780.6	18190.05	18199.41	18471.54	18554.03	17044.053	19308.67	19680.6	20191.14	18121.08	18186.81	18691.193	19684.44	18070.64	18199.58	23267.49	23489.26	21316.32	21634.24	16293.987	16030.04	18306.78	18348.06	23613.34	23535.91	20963.36	21577.28	16194.07	16008.51	18876.4	18570.49
Meters Agv2	19379.7141	19747.44	18197.95	18180.78	18739.03	18321.31	17333.064	18716.83	19817.1	20109.04	18080.69	18474.42	18556.854	19693.8	18392.48	18396.15	23189.4	23431.67	21558.82	21562.91	16416.371	16029.57	18275.33	18216.22	23250.29	23607.75	21272.1	21422.27	16019.28	16008.39	18741.88	18468.83
Meters Agv3	19506.9534	19809.91	18231.55	18098.47	18601.44	16958.18	17468.846	18885.83	19843.76	20078.9	18049.13	20550.64	18322.224	19803.9	18021.52	18382.99	22998.88	23501.89	21125.59	21480.88	16290.067	15977.36	18659.23	18572.22	23261.78	23749.04	21337.18	21680.66	16120.26	15916.94	18965.32	18536.02
Meters Agv4	5.87302	5.98918	5.7137	5.73634	18527.34	16686.63	17310.707	18715	5.90496	6.1131	5.661	5.60296	18747.306	19436.37	17877.72	18506.02	6.8452	6.89766	6.46612	6.49422	16237.394	15918.04	18398.43	18174.74	6.899	6.87072	6.31178	6.45284	16002.5	16181	18606.24	18192.94
Meters Agv5	5.86082	6.00868	5.73662	5.6982	18661.03	15019.19	17647.403	18847.75	5.95348	6.07436	5.66606	5.72776	18878.503	20058.16	17712.32	18566.47	6.76768	6.8583	6.53906	6.50516	16147.204	15896.94	18402.63	17810.71	6.79144	6.92774	6.3772	6.40702	16061.14	15795.97	18495.41	18459.23
Meters Agv6	5.9152	6.00462	5.78144	5.6817	18993.88	10497.06	17548.282	18736.45	5.95538	6.06774	5.65572	5.79876	18616.411	19837.73	18026.23	18191.68	6.7358	6.88736	6.40394	6.47754	16226.479	15904.78	17996.63	18593.15	6.7832	6.95814	6.38948	6.4633	15983.06	15953.94	18746.38	18473.37
Energy consumption Agv1					5.56988	5.65754	5.7382	6.08696					5.57818	5.8192	5.6056	5.67614					4.74774	4.71606	5.55698	5.53078					4.6919	4.66754	5.72532	5.50988
Energy consumption Agv2					5.67892	5.57314	5.67384	5.86694					5.54096	5.81356	5.76916	5.70858					4.80682	4.69574	5.54274	5.53008					4.7057	4.6531	5.65184	5.55838
Energy consumption Agv3					5.634	5.17464	5.68676	5.92108					5.47148	5.8537	5.65752	5.70196					4.7513	4.6707	5.66612	5.61948					4.6853	4.63154	5.7044	5.57306
Energy consumption Agv4					5.6159	5.08646	5.71478	5.87742					5.61248	5.75644	5.63686	5.72248					4.7366	4.67882	5.57318	5.52322					4.75856	4.73348	5.58358	5.48688
Energy consumption Agv5					5.65642	4.58028	5.62002	5.9439					5.62526	5.94548	5.5736	5.76164					4.71106	4.68138	5.58002	5.57478					4.68592	4.60412	5.593	5.59988
Energy consumption Agv6					5.75856	3.19006	5.71954	5.93626					5.57616	5.85342	5.6586	5.70274					4.73784	4.67198	5.4786	5.65084					4.63708	4.65744	5.6302	5.533

6.1.1 Analysis of a single storage policy: Throughput

The diagram shows the throughput obtained by running all 32 simulations under the Random storage policy. The structure of the warehouse under each simulation is highlighted right below the graph and the different colours in the dots identify other variations, this time not in rack configuration but in interarrival time and number of SKUs. There are eight different warehouse structures. The graph clearly shows that no matter the inter-arrival time and the number of SKUs, an increase in the number of vehicles or an increase in the number of corridors, produces greatest results in terms of throughput. On the other hand, it looks like the increase in the number of levels does not create considerable differences as far as final throughput is concerned. By focusing on the effect of the interarrival time and the number of SKUs, it can be observed that, given a specific warehouse configuration, the simulations with a low number of SKUs have given a higher throughput compared to the ones with a higher number of product types. This is understandable since the higher the number of SKUs, the higher the complexity of order fulfilment. The number of orders that can be satisfied in a unit time decreases since more time is allocated to each order. It can be noted that in most of the cases, the scenario with order interarrival time 100 seconds and 5 SKUs has a throughput slightly better compared the scenario with order interarrival time of 50 seconds and 5 SKUs. This might seem quite odd since if the order interarrival time increases, one would expect the throughput to decrease. This does not happen because in these simulations there is the element of disturb which is the presence of retrieval orders. In a configuration with interarrival time equal to 50s there is the creation of orders in the following sequence: six picking orders and one retrieval order. On the other hand, in a configuration with interarrival time equal to 100s there is the creation of orders in the following sequence: three picking orders and one retrieval order. As a consequence, the first case has more picking orders to satisfy in an hour, and since the picking orders take more time to be fulfilled, the overall throughput is lower compared to the other scenario. The situation changes when the throughput is dictated by the rate at which the orders are generated and not by the velocity of vehicles. This is the case of scenario two and four where the blue configuration has a higher throughput compared to the orange one. In the blue configuration, the vehicles never wait for orders to come, the orders are generated very quickly, and they accumulate in a buffer.

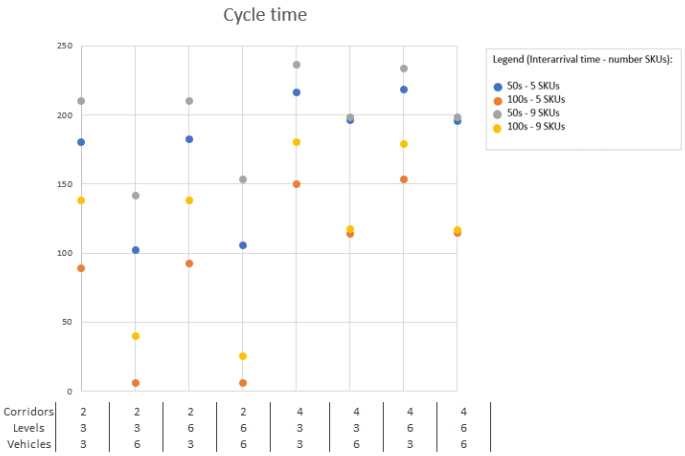
Therefore, the throughput is directly related to the vehicle capacity. In the orange scenario, the vehicles capacity is higher than the rate at which the orders are generated and the maximum throughput that can be obtained is directly linked to the orders arrival rate. In other words, during this simulation, the orders do not wait to be processed at all and the vehicles wait for orders to be generated and consequently less orders are fulfilled. This aspect can be seen in the graphs that show the order waiting times. All the other storage policies share the same trend in the graphs; Therefore, they will not be described.



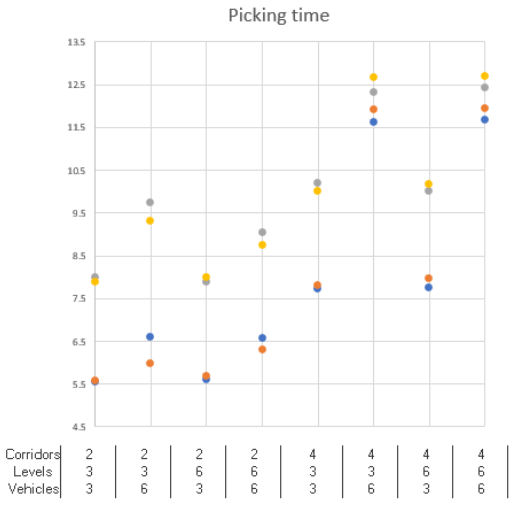
6.1.2 Analysis of a single storage policy: Cycle Time

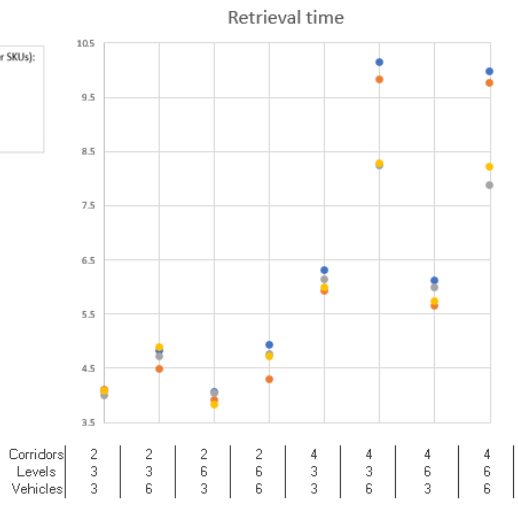
The cycle time has been calculated as the sum of order task time and waiting time. From the respective graph, it is possible to observe that the higher the number of corridors, the higher the cycle time. This happens because whenever the total area of the warehouse increases, the time needed to perform tasks increases as well. The increase in the number of vehicles leads to a decrease in cycle time, because more vehicles are able to satisfy more orders and the waiting time of each order decreases, making the cycle time decrease as

well. Similarly to the throughput, a change in the number of levels does not create great differences in cycle time.



By analysing the data in the cycle time graph, it is clear that the simulations with order interarrival time of 50s and SKUs equal to 5 and 9 are the ones with the highest cycle time. This is due to the high frequency of order interarrival time which makes the orders wait a long time before they can be processed by the vehicles. The waiting time graph confirms that these two scenarios are characterized by a high level of waiting time. On the other hand, if the focus is moved to the Picking time graphs, it is clear that the higher the number of SKUs, the higher the time needed to satisfy the order will be. In fact, scenarios grey and yellow which have SKUs equal to 9 are characterized by a higher picking time compared to scenarios blue and orange, no matter the warehouse structure.



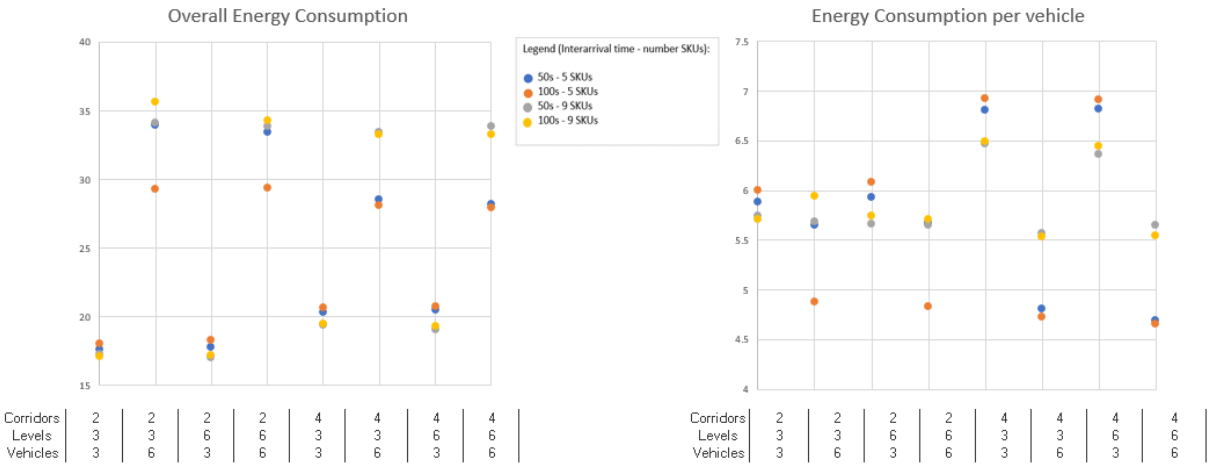


As far as the retrieval time is concerned, a bigger warehouse structure makes the retrieval time increase because the vehicles have to cover more meters to reach a specific slot. This

is particularly evident in the configurations with a higher number of vehicles probably because with more vehicles it is more difficult to find an empty corridor and, in some cases, it is necessary to reach corridors located further away. All the other storage policies share the same trend in the graphs; Therefore, they will not be described.

6.1.3 Analysis of a single storage policy: Energy consumption

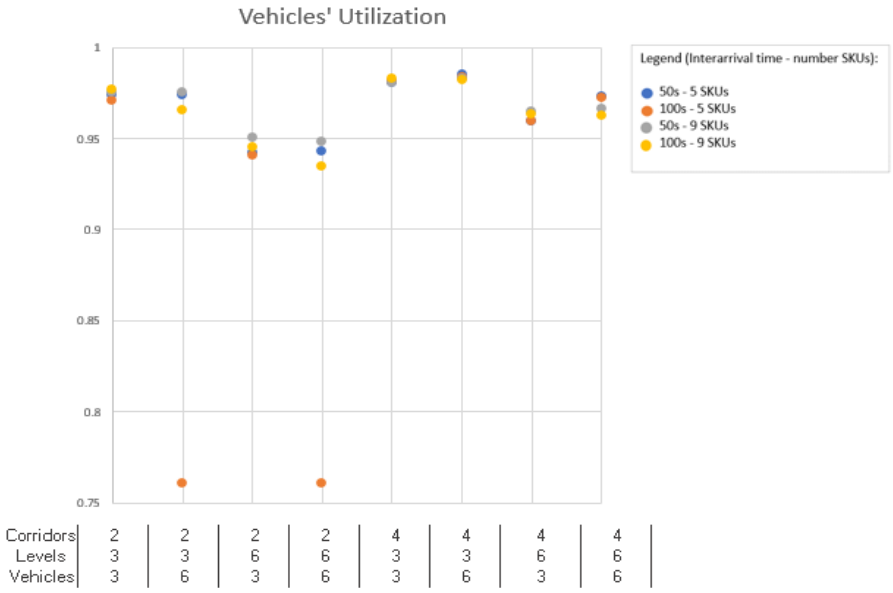
In most of the cases the energy consumption increases as the number of corridors increases and this is justified by the fact that vehicles need to travel more and as a consequence, they consume more. Another aspect that is highlighted in the graph is the behaviour of energy consumption as the number of vehicles increases. No matter the interarrival time and the number of SKUs, whenever the number of vehicles increases, the overall energy consumption increases as well because the vehicles fulfil more orders and perform more tasks. However, in those scenarios where the overall energy consumption of six vehicles is lower than double compared to the scenario with three vehicles, the energy consumption per vehicle of the six vehicles scenarios is lower compared to the three vehicles scenarios. This is justified by the fact that the difference between the overall energy consumptions is not that great and with more vehicles, the energy is spread on a greater number of units, leading to a lower energy consumption per vehicle. An example is given by scenario 3 (2C, 6L, 3V) and 4(2C,6L,6V). In the Overall energy consumption graph, scenario number three has a lower energy consumption compared to scenario four but the energy consumption per vehicle is lower in the scenario four compared to scenario three.



Taking into consideration the number of SKUs it is possible to observe that the energy consumption is higher when both the number of SKUs and the number of vehicles increase. The higher the number of vehicles, the more time vehicles need to wait for a corridor to be freed and the more passive energy is consumed during the waiting time. With a lower number of vehicles, on the other hand, the increase in number of SKUs does not seem to give higher energy consumption values. All the other storage policies share the same trend in the graphs; Therefore, they will not be described.

6.1.4 Analysis of a single storage policy: Vehicle Utilization

As it is possible to infer from the utilization graph, the vehicles' utilization is very high in almost all scenarios except for scenario with 2 corridors, 2 levels, 6 vehicles, 100 s of order interarrival time and 5 SKUs and scenario with 2 corridors, 6 levels, 6 vehicles, 100 s of order interarrival time and 5 SKUs. This is probably due to the fact that in these specific scenarios, the small warehouse structure, the high interarrival time and the low number of SKUs have given vehicles time to fulfil orders without working at full potential during the entire simulation.



Keeping constant the warehouse structure, the increase in the number of vehicles should lead to a decrease in the vehicles' utilization. This is not very clear in the graph because with the two different interarrival times, the orders enter the model quite quickly and the units have no time to rest. However, if the number of vehicles is set to 8 or more or the

interarrival time is increased to 200s or more, this will give as result a lower value in the vehicles' utilization performance variable. Figure 27 proves this by showing graphs with the same rack configuration and with higher number of vehicles first and then with lower values of interarrival time.

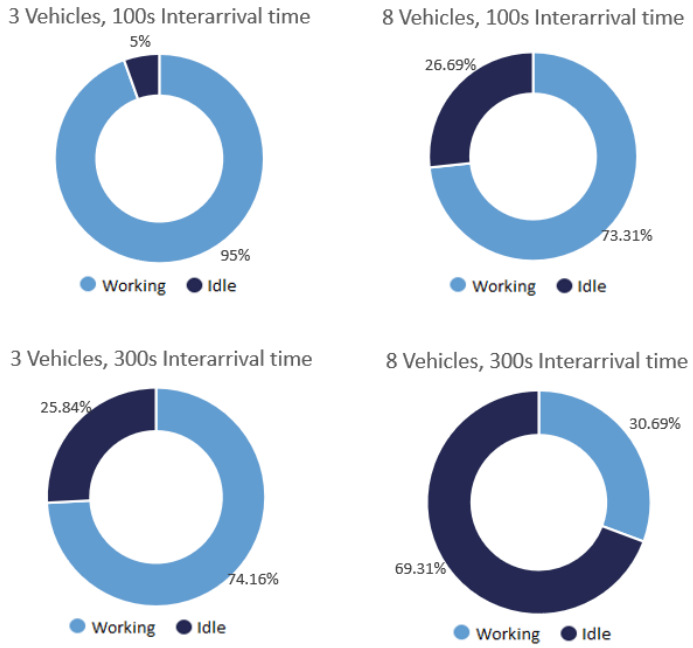


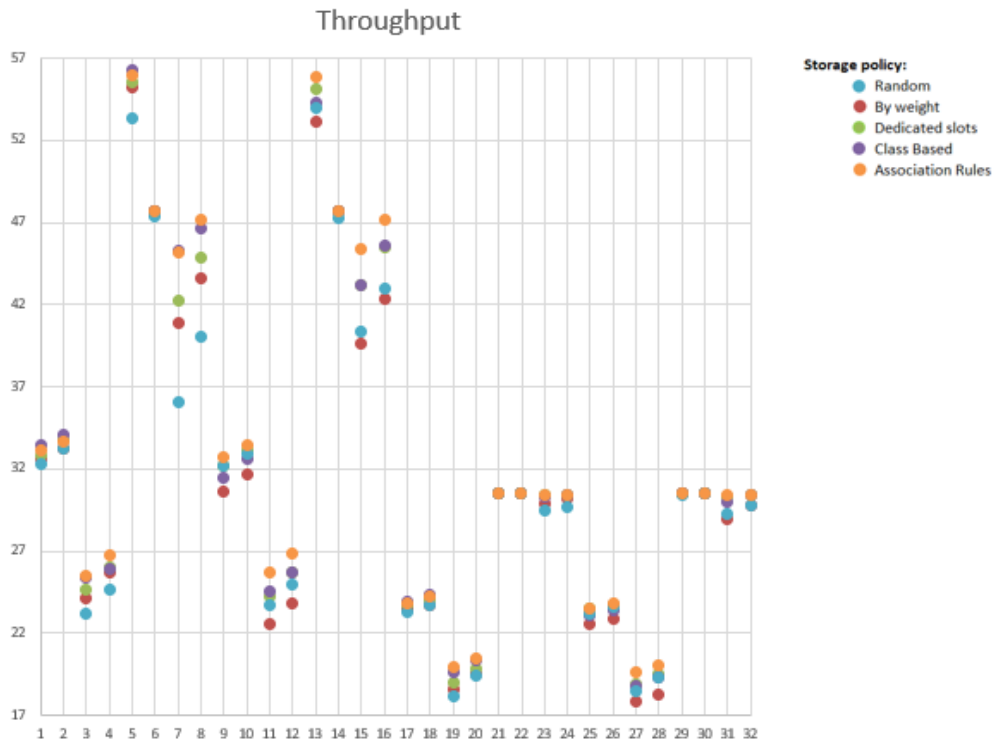
Figure 27: Vehicles' utilization under different number of vehicles and interarrival times

6.1.5 Comparison of the five storage policies

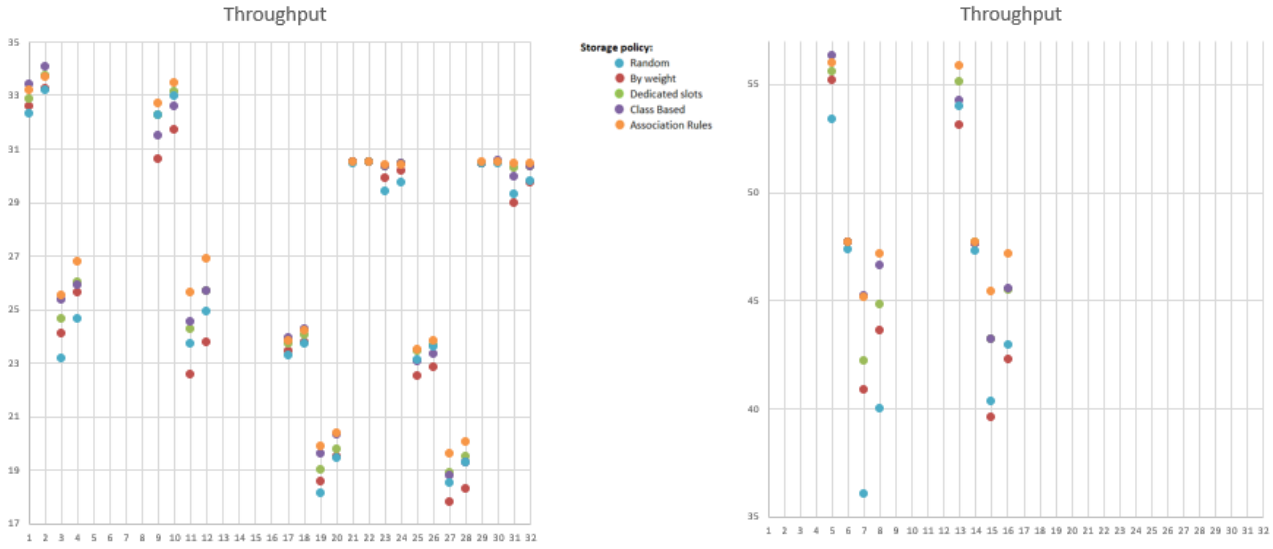
The present study also aims at identifying the best storage policy when it comes to satisfying picking orders through the new innovative technology of Eurofork. The comparison will focus on Throughput, Cycle time and Energy consumption under all 32 simulations.

6.1.5.1 Throughput

The graph below shows the throughput of all five storage policies under the 32 simulations (Table 3). From a first look at the graph it seems like the storage policy with Association rules is the one that gives the best results in almost all scenarios. In order to better understand the data, they will be represented into two different graphs.



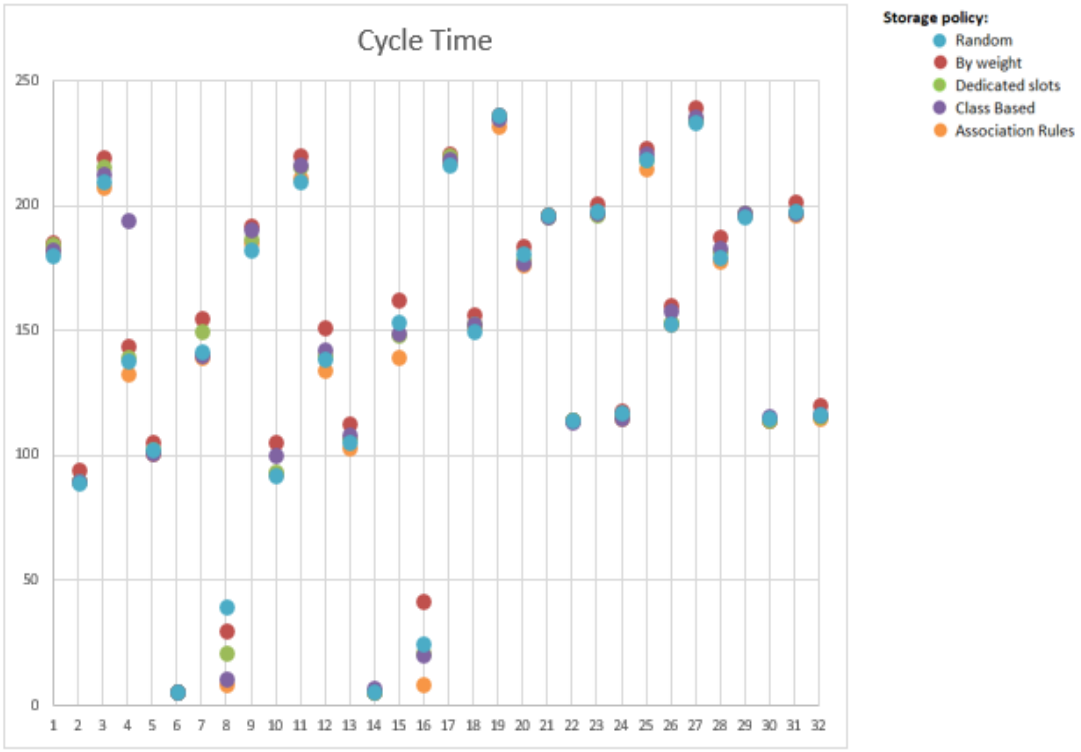
As anticipated, the data points above have been represented onto two separate graphs for clarity reasons. The graph on the right shows the throughput of simulations 5-6-7-8- and 13-14-15-16 whereas the graph on the left shows all the other ones. In most of the cases, the best throughput is given by the storage policy with Association rules. It is relevant to highlight that the throughput of the Class Based storage comes very close to the storage policy with association rules and in some simulations the Class Based Storage policy performs better than the one with Association rules.



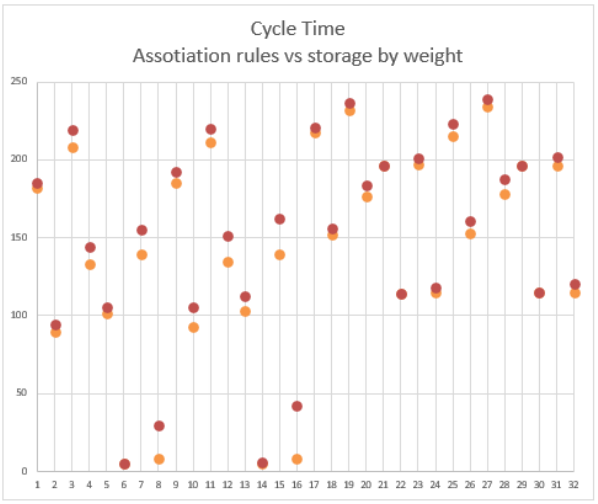
In most of the cases, the worst performing one is the storage by weight policy followed by the random storage. The Dedicated Slots storage shows an average performance.

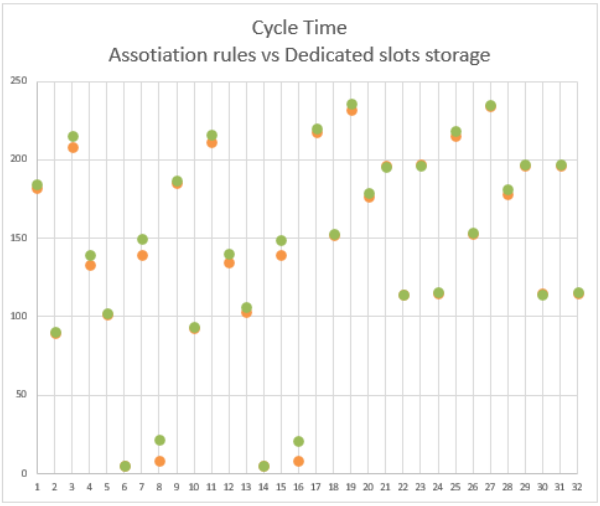
6.1.5.2 Cycle time

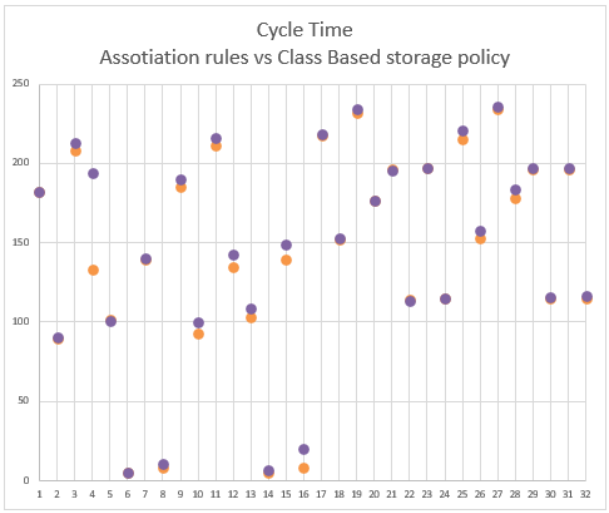
Cycle time is an indication of how much time it takes for an order to be fulfilled and the lower the cycle time, the better. From the data represented in the graph it can be noticed that in many cases the storage policy by Association rules has the lowest Cycle time.

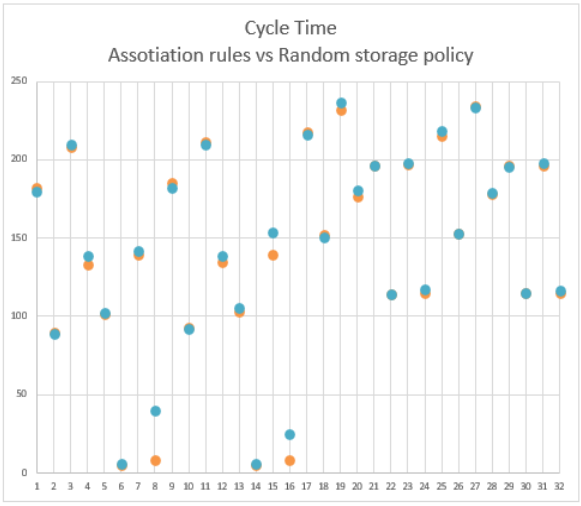


Because in some cases the data are so close to each other that the data points almost overlap, the cycle time of the storage policy by association rules will be compared against the other storage policies. When compared to the storage by weight, it is clear that the cycle time of the storage by association rules is much better. With the other three storage policies, in some cases the storage by association rules is showing a better cycle time and in other cases it is showing a very similar output compared to the others. However, it never happens that the cycle time of the storage by association rules is clearly outperformed by the one of another storage policy.





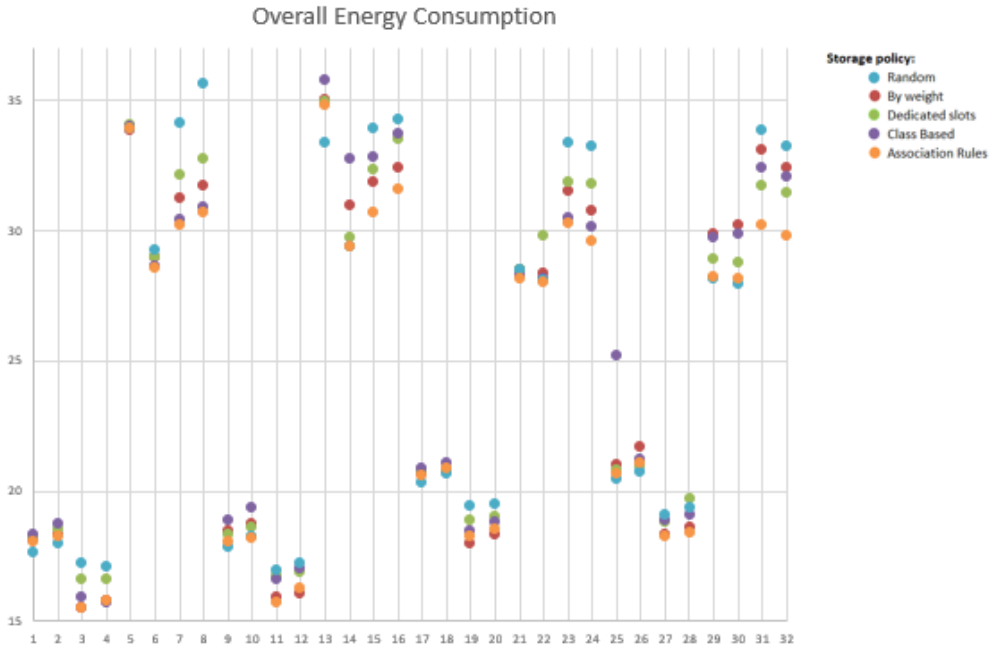




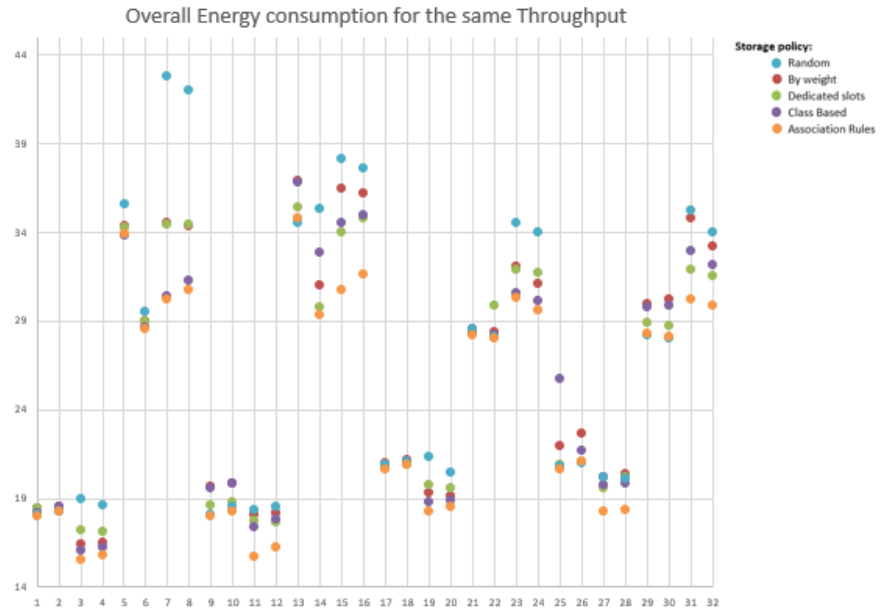
6.1.5.3 Energy consumption

The Energy consumption seems to be lower for the storage policy with association rules across almost all 32 simulations. However, it is important to notice that in some cases, when the number of SKUs is low, like simulations 1-2-17-18, the Random storage policy operates at the lowest energy. In simulations 19-20-12 the policy with the lowest amount of energy consumed is the storage policy by weight. The isolated interpretation of this graph could lead to some wrong statements that believe in the success of the random policy and the storage by weight in certain simulations. A better interpretation would be to study the energy graph in relation to the throughput generated by the five storage policies and see which one has used energy in the most efficient way. Through a simple proportion, it has been calculated the energy that the different storage policies would

consume if they had to achieve the same throughput as the storage policy by association rules.



Thanks to the new graph, it is possible to appreciate even more the difference that lies among the five storage polices. In most of the cases, the random storage policy and the storage by weight are the ones that consume more energy in total. This aspect was hidden in the previous graph. The dedicated slots storage shows an average to low performance and the Class Based storage policy seems to perform well when the number of SKUs if high, but it performs poorly with a lower number of SKUs.



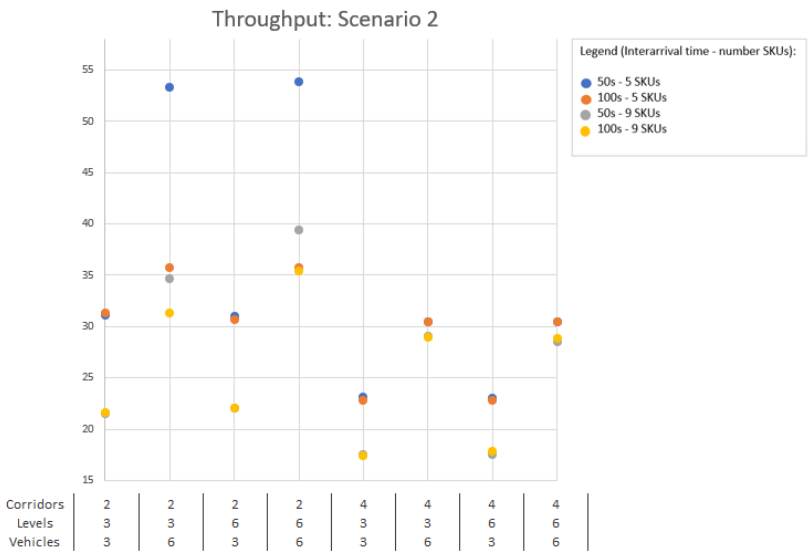
This is probably due to the fact that with lower SKUs the different types of products are well spaced apart, and it might require more energy to fulfil an entire order.

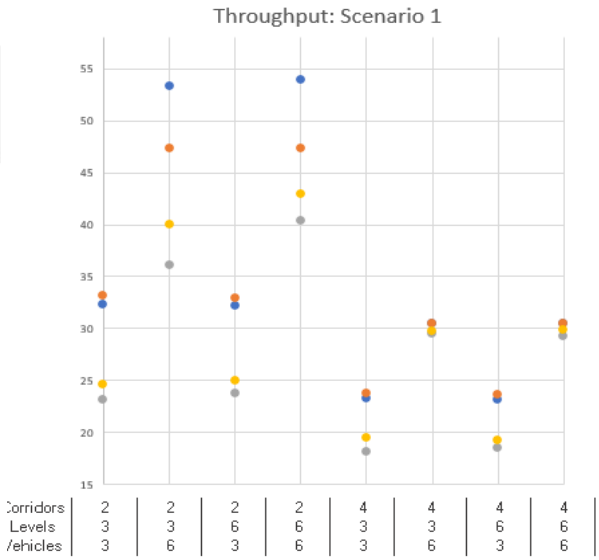
6.2 Scenario 2: Picking orders only

In this second scenario, the retrieval activity has been removed so that the only task assigned to the vehicles is the picking order task. Generally speaking, the overall trends of the graphs do not change but the numerical value of the single simulations varies. The results obtained during the analysis of the second scenario, will be directly compared to the results obtained from scenario 1. Similarly to what has been done previously, the Random storage policy will be analysed independently, since its results have the same trend of the other policies, and then the five storage policies will be compared to see which one is the best under scenario2.

6.2.1 Analysis of a single storage policy: Throughput

The removal of the retrieval activity did not change the throughput trend already explained in Scenario 1. What can be observed from the two scenarios is that in the majority of the cases the throughput has decreased, sometimes the decrease is more evident and in other simulations it is more subtle. The decrease is justified by the fact that in scenario 2 there are picking orders only which generally need more time to be processed.



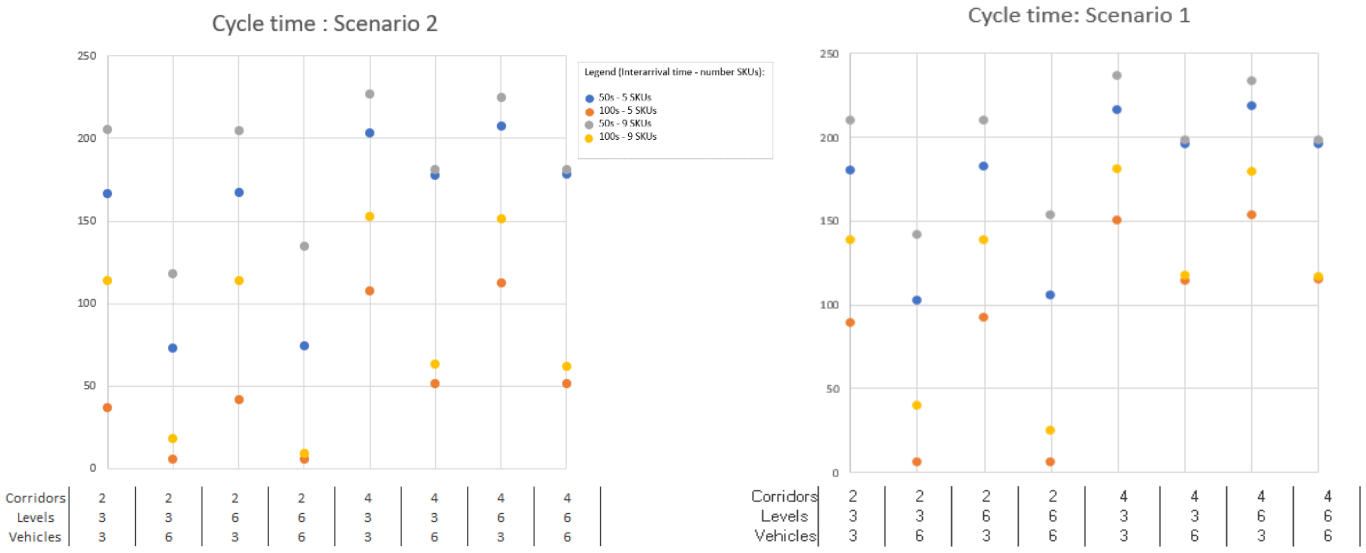


The retrieval orders, on the other hand are quicker to process and they contribute to increase the throughput in scenario 1.

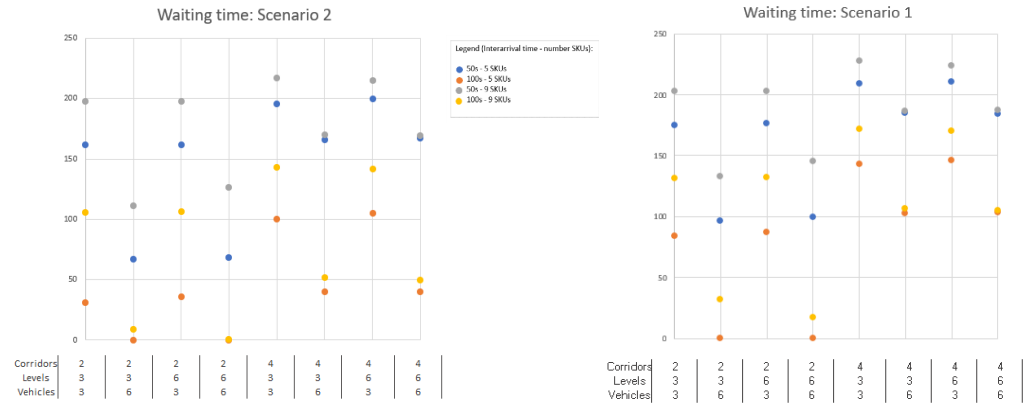
Another difference worth highlighting is that, in scenario 2 the simulations which differ only from the interarrival time have a very similar throughput. This happens because, no matter the two interarrival times, there are no retrieval orders (quicker orders) and the throughput is given by the capacity of the vehicles and it does not change when the interarrival time changes from 50 s to 100s. The difference between throughput can be observed in simulation two and four in the graphs, but this is due to the fact that in one case the throughput is determined by the interarrival time (blue data point) and in the other case the throughput is determined by the vehicles' capacity (orange data point).

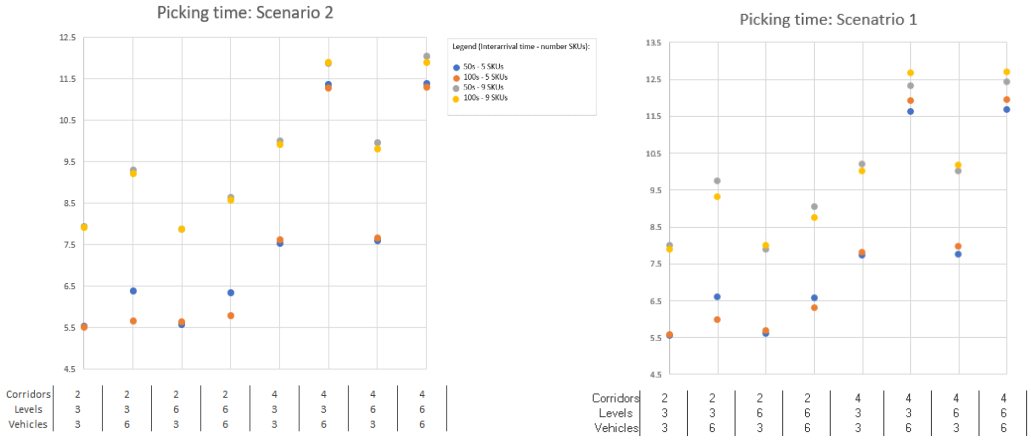
6.2.2 Analysis of a single storage policy: Cycle time

When observing the Cycle time graphs it is clear that the scenario with picking orders only, Scenario 2, is characterized by a lower cycle time in almost all simulations. This happens because, keeping everything else fixed, the number of incoming orders is lower and consequently the order waiting time is lower, this makes the cycle time decrease.



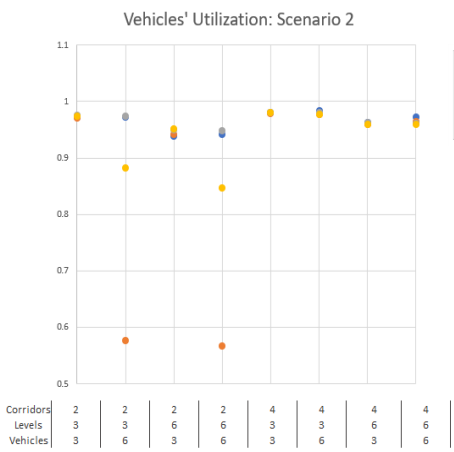
The picking time shows a very similar behaviour compared to scenario 1 while the waiting time seems to be much lower compared to scenario 1 in almost all simulations.

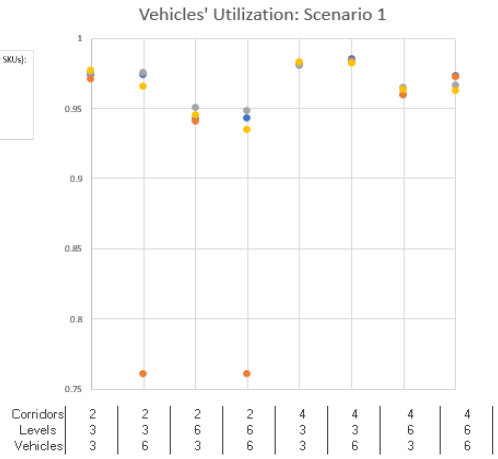




6.2.3 Analysis of a single storage policy: Energy Consumption

The analysis of the energy consumption confirms what already discovered in the previous graphs. The lower number of incoming orders shows a lower value of energy consumed because vehicles have overall less orders to process. In addition to that, also the vehicles' utilization is lower in the second scenario compared to the first one.

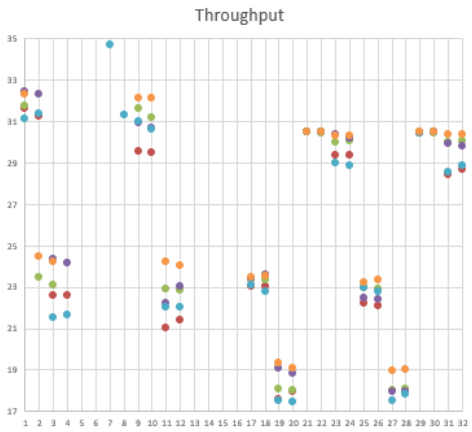


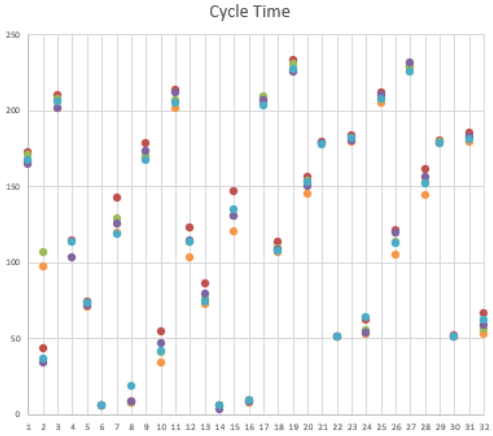


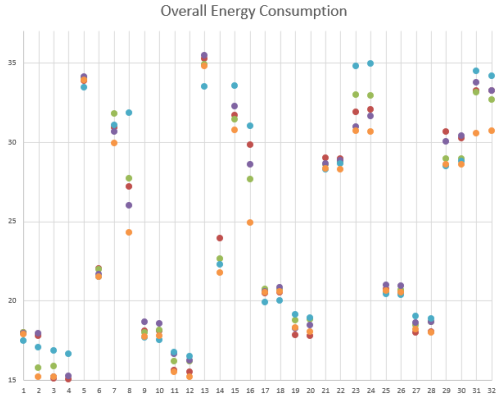
It happens more frequently that vehicles wait for orders to arrive rather than orders waiting for vehicles to be available.

6.2.4 Comparison of the five storage policies

The comparison of the five storage policies of scenario 2 gives the same results as the comparison of the storage policies under scenario 1. This suggests that the retrieval activities with an order interarrival time of 300s does not influence on the final results. The storage policy that gives the best outcomes in terms of throughput, cycle time and energy consumption is the storage policy by association rules.





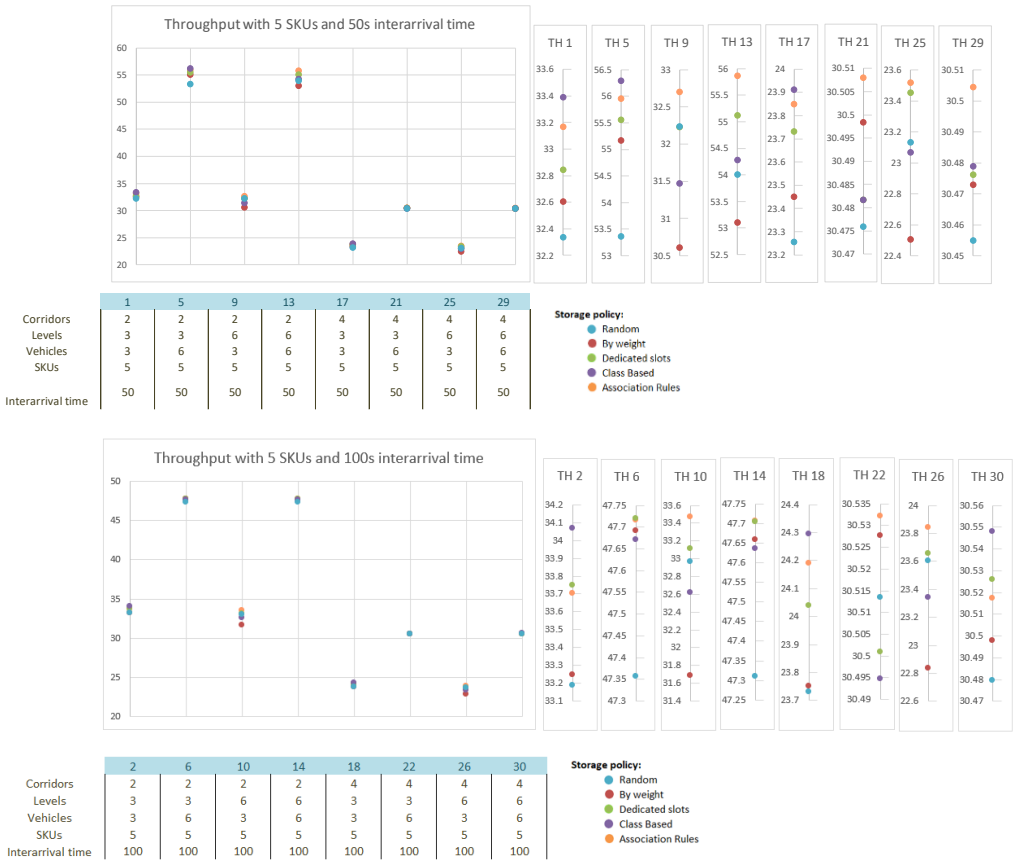


7. Final considerations on results

The following section will investigate in more details the results obtained from the analysis of Scenario 1 to understand what the best storage policy for every warehouse configuration is. This last analysis is important because it completes the results already discussed, which were approximative and quite vague, the past results presented a description of the data and they suggested that the storage by association rules is the best policy. The present analysis goes deeper to see if that storage policy is really the best one in every scenario considering all performance variables at the same time. Only Scenario number 1 will be considered as it shows a dynamic closer to what happens in reality, with both retrieval and picking orders.

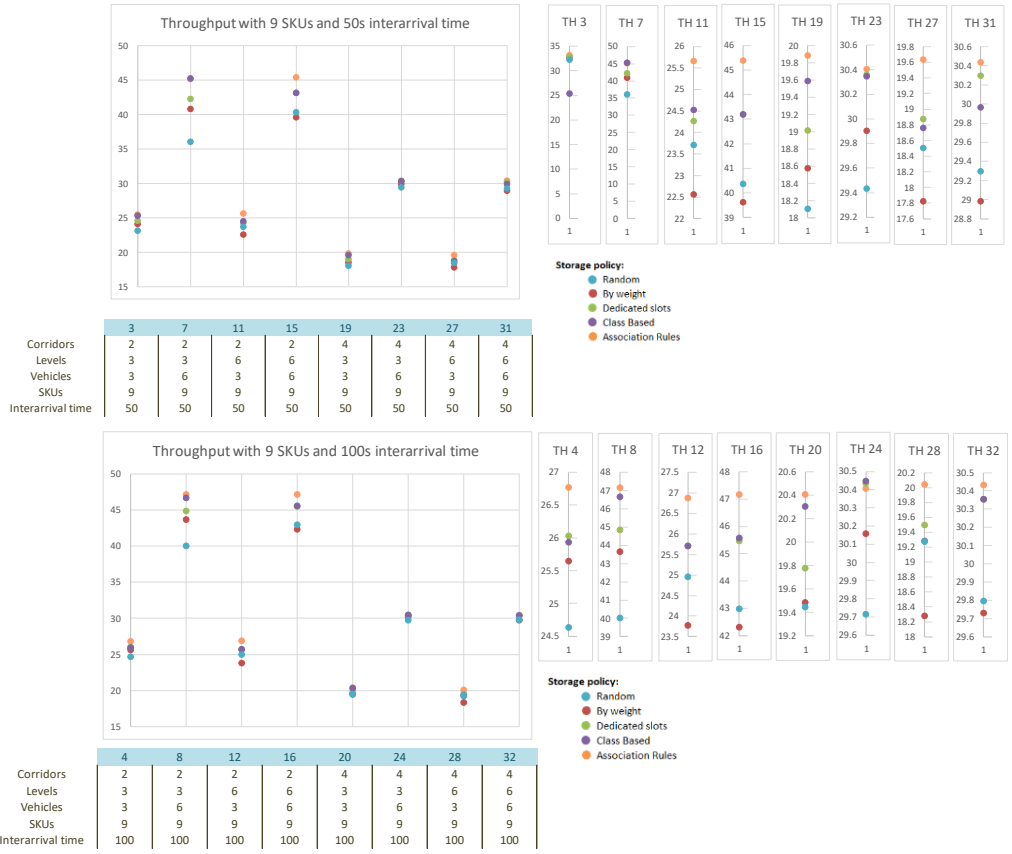
The following graphs show how the different policies performed in terms of throughput, cycle time and energy consumption. For every performance variable, there are four graphs which represent 32 different results based on SKUs and order interarrival time variations.

For every representation, the smaller graphs on the right are simply a zoomed view of the graph on the left. As far as throughput is concerned, the highest the result, the better it is.



	1	5	9	13	17	21	25	29
Corridors	2	2	2	2	4	4	4	4
Levels	3	3	6	6	3	3	6	6
Vehicles	3	6	3	6	3	6	3	6
SKUs	5	5	5	5	5	5	5	5
Interarrival time	50	50	50	50	50	50	50	50

	2	6	10	14	18	22	26	30
Corridors	2	2	2	2	4	4	4	4
Levels	3	3	6	6	3	3	6	6
Vehicles	3	6	3	6	3	6	3	6
SKUs	5	5	5	5	5	5	5	5
Interarrival time	100	100	100	100	100	100	100	100

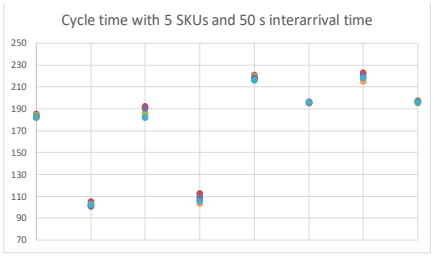


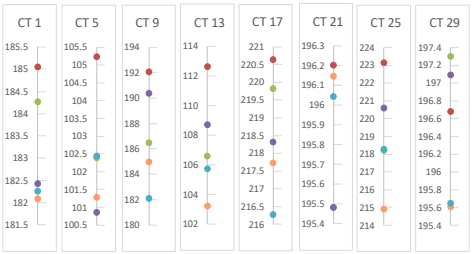
	3	7	11	15	19	23	27	31
Corridors	2	2	2	2	4	4	4	4
Levels	3	3	6	6	3	3	6	6
Vehicles	3	6	3	6	3	6	3	6
SKUs	9	9	9	9	9	9	9	9
Interarrival time	50	50	50	50	50	50	50	50

	4	8	12	16	20	24	28	32
Corridors	2	2	2	2	4	4	4	4
Levels	3	3	6	6	3	3	6	6
Vehicles	3	6	3	6	3	6	3	6
SKUs	9	9	9	9	9	9	9	9
Interarrival time	100	100	100	100	100	100	100	100

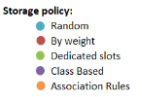
It can be observed that the Class Based storage policy could be a good solution especially when the structure of the warehouse is small, and the number of SKUs are low. In fact, the highest throughput is given by Th Class Based storage policy in simulations 1, 2, 5, 7 where the number of levels and corridors is at its minimum. When the number of SKUs is small, the Class based storage can be a good option even in a slightly bigger area, where the number of corridors is 4 and the other parameters are at their minimum. However, this does not hold true when the number of SKUs increases. For all other scenarios, the storage policy by association rules is the best one, except for scenario 6 where the best one seems to be the dedicated slots policy.

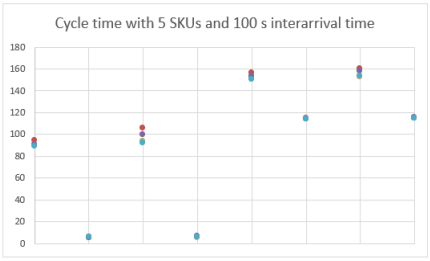
The next graphs show in detail the cycle time results for all five storage policies. Contrarily to what happens with throughput, when it comes to cycle time, the lowest value it has, the better the policy is performing. Therefore, the best policies will be the ones that rank the lowest in the graphs.

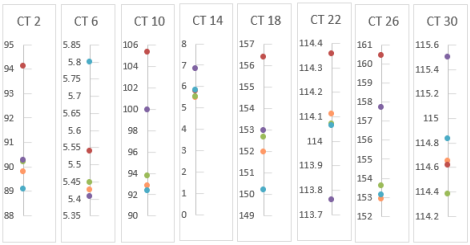




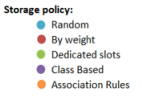
	1	5	9	13	17	21	25	29
Corridors	2	2	2	2	4	4	4	4
Levels	3	3	6	6	3	3	6	6
Vehicles	3	6	3	6	3	6	3	6
SKUs	5	5	5	5	5	5	5	5
Interarrival time	50	50	50	50	50	50	50	50

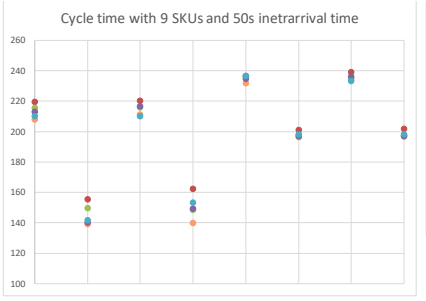


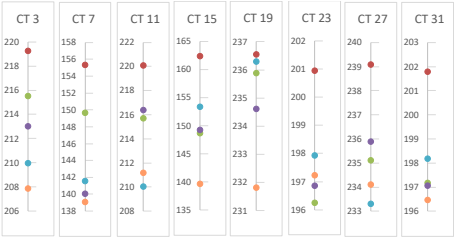




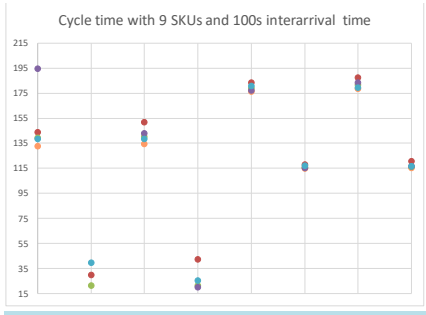
	2	6	10	14	18	22	26	30
Corridors	2	2	2	2	4	4	4	4
Levels	3	3	6	6	3	3	6	6
Vehicles	3	6	3	6	3	6	3	6
SKUs	5	5	5	5	5	5	5	5
Interarrival time	100	100	100	100	100	100	100	100

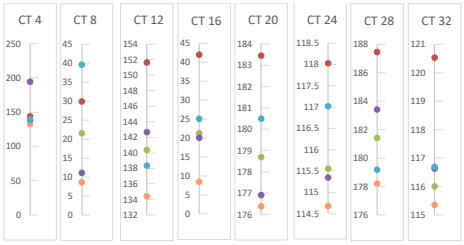




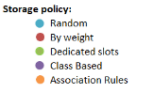


	3	7	11	15	19	23	27	31
Corridors	2	2	2	2	4	4	4	4
Levels	3	3	6	6	3	3	6	6
Vehicles	3	6	3	6	3	6	3	6
SKUs	9	9	9	9	9	9	9	9
Interarrival time	50	50	50	50	50	50	50	50





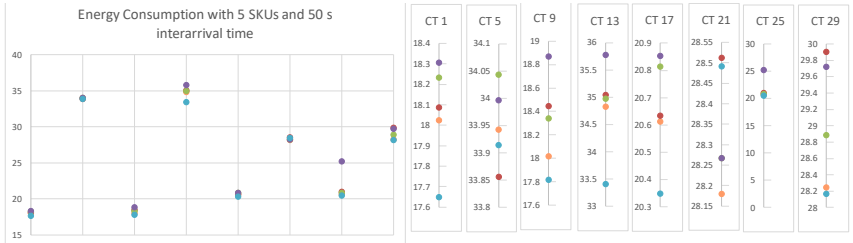
	4	8	12	16	20	24	28	32
Corridors	2	2	2	2	4	4	4	4
Levels	3	3	6	6	3	3	6	6
Vehicles	3	6	3	6	3	6	3	6
SKUs	9	9	9	9	9	9	9	9
Interarrival time	100	100	100	100	100	100	100	100



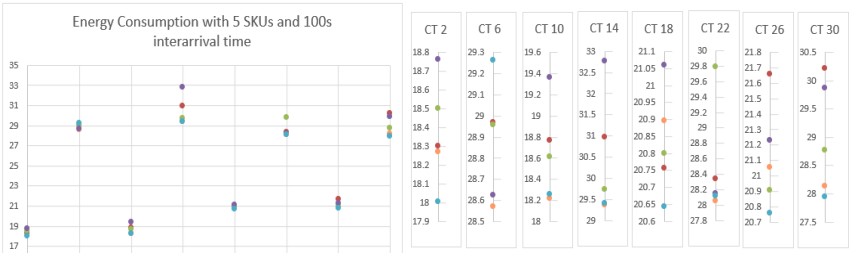
The results obtained with the analysis of cycle time show different results compared to what observed with the throughput. This mean that looking at the cycle time only, other storage policies seem to be the best ones.

The Random storage policy seems to be the best option especially in those simulations in which there is a slightly variation in the dimensions on the warehouse in combination with a low SKU value. In fact, simulations 9-10-17-18 show exactly what just observed. The random storage policy could also be a solution when the number of SKUs is equal to 9, however it can be observed that its performance decreases as the number of SKUs increases. The class-based storage policy is the best option when the number of vehicles is at its maximum and the structure of the warehouse is small or relatively small like simulations 5-6-21-22. The Dedicated slots storage is the best performing policy in simulation 23 and 30 where there is a substantial level of complexity.

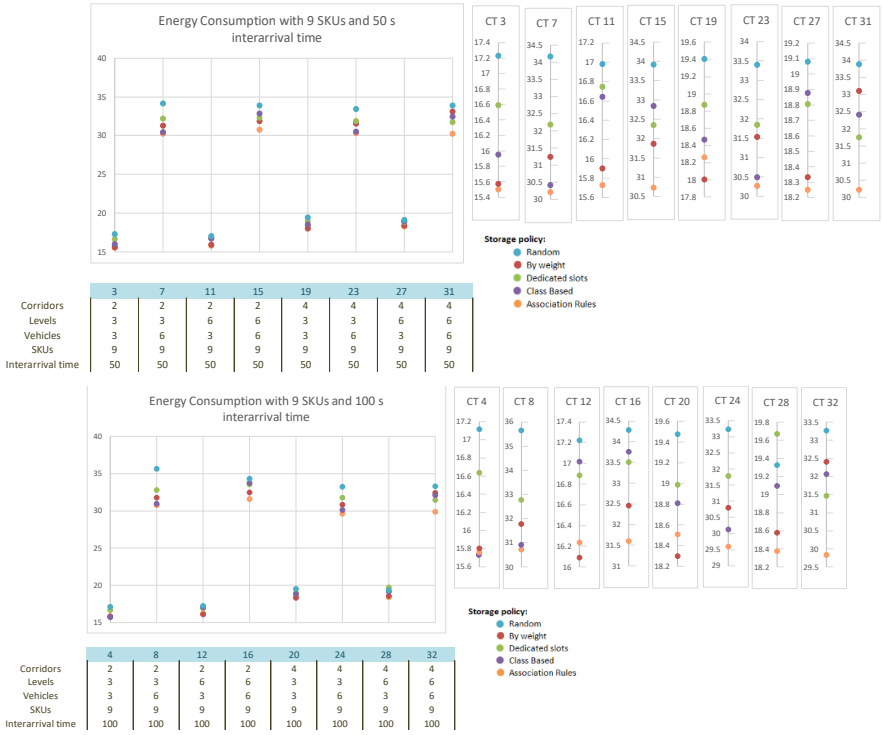
The following graphs show the best performing policies under the perspective of Energy consumption. Similarly to Cycle time, the lower energy is consumed, the better it is.



	1	5	9	13	17	21	25	29
Corridors	2	2	2	2	4	4	4	4
Levels	3	3	6	6	3	3	6	6
Vehicles	3	6	3	6	3	6	3	6
SKUs	5	5	5	5	5	5	5	5
Interarrival time	50	50	50	50	50	50	50	50



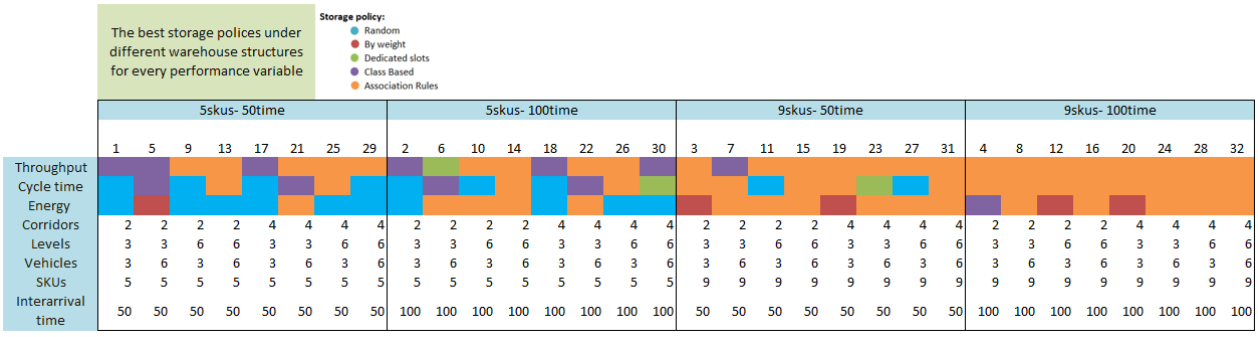
	2	6	10	14	18	22	26	30
Corridors	2	2	2	2	4	4	4	4
Levels	3	3	6	6	3	3	6	6
Vehicles	3	6	3	6	3	6	3	6
SKUs	5	5	5	5	5	5	5	5
Interarrival time	100	100	100	100	100	100	100	100



	3	7	11	15	19	23	27	31
Corridors	2	2	2	2	4	4	4	4
Levels	3	3	6	6	3	3	6	6
Vehicles	3	6	3	6	3	6	3	6
SKUs	9	9	9	9	9	9	9	9
Interarrival time	50	50	50	50	50	50	50	50

	4	8	12	16	20	24	28	32
Corridors	2	2	2	2	4	4	4	4
Levels	3	3	6	6	3	3	6	6
Vehicles	3	6	3	6	3	6	3	6
SKUs	9	9	9	9	9	9	9	9
Interarrival time	100	100	100	100	100	100	100	100

The Random storage policy shows low values in energy consumption when the number of SKUs is low and the structure of the warehouse is relatively small. On the other hand, when the number of SKUs increases, the Random storage policy shows poor performances in energy consumption. In these scenarios the storage policies by weight and by association rules have the best outcomes. The storage policy by weight has low energy consumptions in simulations 5-19-12-20 where the structure of the warehouse is relatively small and the number of vehicles is low. In all other cases, the storage policy by association rules is the best one in terms of energy consumption. Figure 28 summarizes what discussed above, highlighting what is the best storage policy for every performance variable under different warehouse structures, number of SKUs and interarrival times.



	1	5	9	13	17	21	25	29	2	6	10	14	18	22	26	30	3	7	11	15	19	23	27	31	4	8	12	16	20	24	28	32
Corridors	2	2	2	2	4	4	4	4	2	2	2	2	4	4	4	4	2	2	2	2	4	4	4	4	2	2	2	2	4	4	4	4
Levels	3	3	6	6	3	3	6	6	3	3	6	6	3	3	6	6	3	3	6	6	3	3	6	6	3	3	6	6	3	3	6	6
Vehicles	3	6	3	6	3	6	3	6	3	6	3	6	3	6	3	6	3	6	3	6	3	6	3	6	3	6	3	6	3	6	3	6
SKUs	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	9	9	9	9	9	9	9	9	9	9	9	9	9	9	9	9
Interarrival time	50	50	50	50	50	50	50	50	100	100	100	100	100	100	100	100	50	50	50	50	50	50	50	50	100	100	100	100	100	100	100	100

Figure 28: The best performing storage policies under different scenarios for different performance variables

The best storage policies are indicated through their respective colour as indicated in the legend: blue is Random, Red is Storage by weight, Green is Dedicated slots storage, Violet is CBS and Orange is storage by association rules.

It can be noted that in some simulations, there is not one single storage policy that outperforms the other ones in all performance variables. In simulation 1 for example the storage policy that gives the best throughput is the Class Based storage, however the one that gives the best cycle time and the best energy consumption is the Random storage policy. In order to conclude the study and to decide the single best storage policy that should be used in the different simulations, an AHP analysis will be carried out.

7.1 AHP Analysis on results

The Analytical Hierarchy Process is a multi-criteria decision-making tool that enables to select the best option out of several alternatives which are all evaluated based on some shared criteria (Pachemska T.A. et al., 2014). The AHP analysis has been developed for those simulations that did not agree on the best storage policy, in all those cases, the problem has been developed using a hierarchical structure. Figure 29 exemplifies the hierarchical structure for the AHP analysis under simulation 1. The fist level of the graph represents the Objective of the study, the second layer represents the criteria under which the options are evaluated, and the last level shows the alternatives available. The AHP methodology is quite flexible, and it is based on the Saaty scale (Figure 30) to explain the relationship among criteria and between criteria and alternatives (Pachemska T.A. et al., 2014). This is an instrument that has 9 grades for intensity of importance.

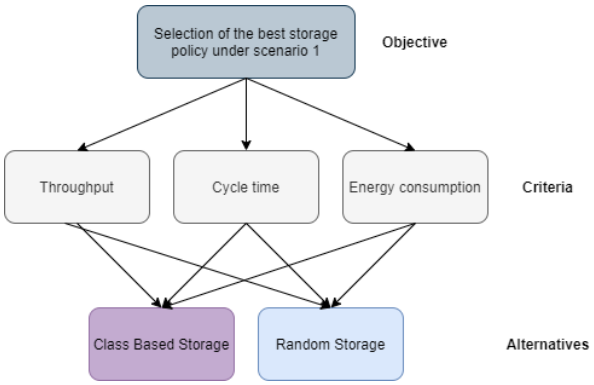


Figure 29: Hierarchical structure for AHP analysis on simulation 1

Intensity of importance	Definition	Explanation
1	Equal Importance	Two activities contribute equally to the objective
2	Weak or Slight	
3	Moderate Importance	Experience and judgment slightly favor one activity over another
4	Moderate Plus	
5	Strong Importance	Experience and judgment strongly favor one activity over another
6	Strong Plus	
7	Very Strong	An activity is favored very strongly over another
8	Very, very Strong	
9	Extreme Importance	The evidence favoring one activity over another is of the highest possible order of affirmation

Figure 30: Saaty's scale (Alireza Afshari et al., 2010)

The preferences of the criteria have been inserted considering that the ability to fulfil as many orders as possible is certainly an ability appreciated by the customer and that lower cycle time and lower energy consumption alone do not automatically equal to a successful outcome if the aim is to satisfy incoming orders. Therefore, the importance of the criteria has been given as it is showed below.

Comparison matrix of 3 criteria			
	Throughput	Cycle time	Energy consumption
Throughput	1	6	5
Cycle time	0.166666667	1	2
Energy consumption	0.2	0.5	1

The results of the AHP analysis are showed in Table 9 and Figure 31 summarizes the AHP results next to the results previously showed.

Table 9: AHP results for every simulation

Simulation	Alternatives	Result	Rank
1	CBS	0.504196499	1
	Random	0.495803501	2
5	CBS	0.505294791	1
	Weight	0.494705209	2
9	Association rules	0.501623039	1
	Random	0.498376961	2
13	Association rules	0.505909609	1
	Random	0.494090391	2
17	CBS	0.503911743	1
	Random	0.496088257	2
21	CBS	0.499901232	2
	Association rules	0.500098768	1
25	Association rules	0.503359121	1
	Random	0.496640879	2
29	Association rules	0.500131011	1
	Random	0.499868989	2
2	CBS	0.503012512	1
	Random	0.496987488	2
6	CBS	0.333433603	1
	Random	0.333219286	3
	Association rules	0.333347112	2
10	Association rules	0.502505916	1
	Random	0.497494084	2
18	CBS	0.502882829	1
	Random	0.497117171	2
22	Association rules	0.500090348	1
	CBS	0.499909652	2
26	Association rules	0.501488	1
	Random	0.498512	2
30	CBS	0.331995562	3
	Dedicated slots	0.334084887	1
	Random	0.33391955	2
3	Association rules	0.512281836	1
	Weight	0.487718164	2
7	CBS	0.499861323	2
	Association rules	0.500138677	1
11	Association rules	0.516053364	1
	Random	0.483946636	2
19	Association rules	0.505806697	1
	Weight	0.494193303	2
23	Association rules	0.502812468	1
	Dedicated slots	0.497187532	2
27	Association rules	0.527337219	1
	Random	0.472662781	2
4	CBS	0.478492265	2
	Association rules	0.521507735	1
12	Association rules	0.526827441	1
	Weight	0.473172559	2
20	Association rules	0.509628337	1
	Weight	0.490371663	2

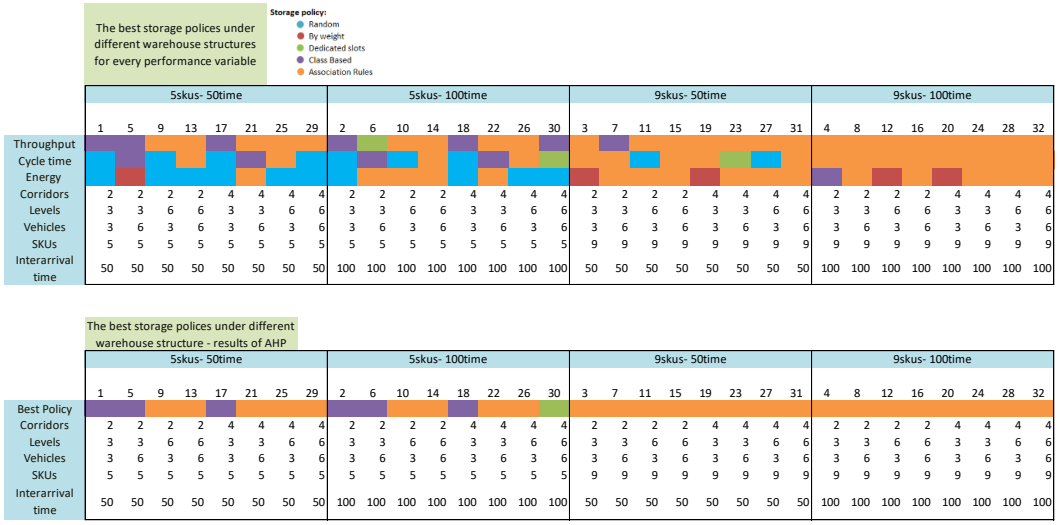


Figure 31: The best performing storage policies under different scenarios for different performance variables (top) and the best storage policies under different scenarios after AHP analysis (bottom)

The AHP analysis put some clarity on which storage policy is the best one under the scenarios studied and in combination with the new technology. In those cases where the warehouse structure is relatively small, the AHP analysis highlights that the CBS policy is the best one and this holds true for simulations 5,17,2,6,18 where the level of complexity of the structure is still quite low. However, as the complexity of the structure increases and especially as the number of SKUs increases, it is clear that the storage policy by association rules offers greater benefits compared to all the other options.

8. Conclusions

The present study wanted to evaluate the performance of the new technology of Eurofork consisting of an AVS/RS system operating though vehicles composed by a shuttle, a satellite and a robotic arm mounted on the shuttle. The study evaluated the behaviour of the technology when dealing with picking activity under different storage policies. The aim of the thesis was to generally understand the performance of the technology under different rack configurations and different scenarios as well as finding the most efficient storage policy that gives the best outcomes when combined with the innovative technology. The study has been performed through Discrete Event Simulation, using FlexSim as simulation software. The results of the 1600 simulations show that the output in terms of throughput is responsive when the physical structure of the facility changes. Generally, a smaller warehouse facilitates the fulfilment of a higher number of orders with low cycle time and relatively low energy consumption. The same reasoning applies when increasing the number of vehicles. The simulations also pointed out that a storage policy by weight is not the most efficient one when it comes to picking activities, contrary to what initially believed and the best storage policy to use in combination with the new shuttle and robotic arm is the storage policy by association rules. This one often outperforms all the other storage policies on all fronts studied: throughput, cycle time and energy consumption. The study of scenario 2, which excludes the retrieval activities from the simulations, does not change the final outcome: according to this study the storage policy by association rules is still the best performing policy in combination with the innovative technology. The final analysis performed with the AHP methodology, applied for that data in scenario1, confirmed the results already discussed in the preliminary analysis. The AHP removed all the doubts in identifying the best storage policy in those simulations where different policies were considered the best under different performance variables. The outcome proves that although the Class Based Storage policy is great in those simulations with a lower level of complexity, the best one in more complex, and therefore closer to reality, scenarios is the storage policy by association rules.

The present study shows some limitations as it is very difficult to replicate what happens in real life in a simulation model and the model itself does not contain all the complexities

that may characterize a real warehouse. The present study could be further expanded by analysing other storage policies or by studying how the different storage policies behave under scenarios more similar to reality. An interesting further analysis would be to simulate the behaviour of the innovative technology on a real case scenario rather than a generic warehouse like the one proposed in this thesis. Despite these limitations, the thesis still gives a good understanding of what are the performances of the technology in an automated warehouse under different scenarios and storage policies.

9. References

AKPUNAR A., EKREN B. Y. & LERHER T. 2017. Energy efficient design of autonomous vehicle based storage and retrieval system. *Istrazivanja i projektovanja za privredu. ,* 15**,** 25-34.

ALIREZA AFSHARI, MAJID MOJAHED & YUSUFF, R. M. 2010. Simple Additive Weighting approach to Personnel Selection problem. *International Journal of Innovation, Management and Technology,* 1.

AURELIJA., B. 2010. Order picking process at warehouses. *Int. J. Logistics Systems and Management,* 6.

BAGUI, S. 2019. Positive and negative association rule mining in Hadoop's MapReduce environment. *Journal of Big Data,* 6.

BRUNO G., D'ANTONIO G. & DE MADDIS M. 2016. Sustainability Analysis of Autonomous Vehicle Storage and Retrieval Systems. *WSEAS Transactions on Environment and Development,* 12**,** 299-306.

COLLA V. & NASTASI G. 2010. Modelling and Simulation of an Automated Warehouse for the Comparison of Storage Strategies. *Modelling, Simulation and Optimization, IntechOpen*.

DARIA BATTINI, MARTINA CALZAVARA, ALESSANDRO PERSONA & SGARBOSSA, F. 2016. Dual-tray Vertical Lift Modules for fast Order Picking. *IMHRC Proceedings (Karlsruhe, Germany),* 6.

EDER M 2020. An approach for a performance calculation of shuttle-based storage

and retrieval systems with multiple-deep storage. *The International Journal of Advanced Manufacturing Technology,* 107**,** 859-873.

EDER M. & KARTNIG G. 2016. Throughput analysis of S/R shuttle system and ideal geometry for high performance. *FME Transactions,* 44**,** 174-179.

EKREN B. Y. & HERAGU 2011. Simulation based performance analysis of an autonomous vehicle storage and retrieval system. *Simulation Modelling Practice and Theory,* Vol 19**,** 1640-1650.

EKREN B. Y. & HERAGU 2012. Performance comparison of two material handling systems: AVS/RS and CBAS/RS. *International Journal of Production Research,* 50:15**,** 4061-4074.

EKREN B.Y., SARI, Z. & T., L. 2015. Warehouse Design under Class-Based Storage Policy of Shuttle-Based Storage and Retrieval System. *IFAC-PapersOnLine,* 48(3)**,** 1152-1154.

FABIO SGARBOSSA, MARTINA CALZAVARA & PERSONA, A. 2019. Throughput models for a dual-bay VLM order picking system under different configurations. *Industrial Management & Data Systems,* 119**,** 1268-1288.

FARAH HANANI M.K., ZULKHAIRI M.Y., MOHAMAD ZAIHIRAIN M.R., MOHD ASWADI A. & A., I. 2016. Development of Automated Storage and Retrieval System (ASRS) for Flexible Manufacturing System (FMS). *Journal of Engineering Technology,* 4**,** 43-50.

FLEXSIM. *OVERVIEW OF 3D LIBRARY OBJECTS* [Online]. Available: https://docs.FlexSim.com/en/21.0/Using3DObjects/Overview3DObjects/ [Accessed 02/03/2021].

FLEXSIM. *Welcome to FlexSim* [Online]. Available: https://docs.FlexSim.com/en/21.0/Introduction/Welcome/ [Accessed 01/03/2021].

GINO MARCHET, MARCO MELACINI & PEROTTI, S. 2015. Investigating order picking system adoption: a case-study-based approach. *International Journal of Logistics Research and Applications: A Leading Journal of Supply Chain Management,* 18:1**,** 82-98.

GIULIA BRUNO & D'ANTONIO, G. 2018. Flexible reconfiguration of AVS/RS operations for improved integration with manufaturing processes. *Science Direct,* 78**,** 196-201.

GUERRAZZI E., MININNO V., ALOINI D., DULMIN R., SCARPELLI C. & SABATINI M. 2019. Energy evaluation of deep-lane autonomous vehicle storage and retrieval system. *Sustainability,* 11.

HAU LING CHAN & KING WAH PANG 2011. Association Rule Based Approach for Improving Operation Efficiency in a Randomized Warehouse. *International Conference on Industrial Engineering and Operations Management*.

JENNIFER A, PAZOUR & MELLER, R. D. 2012. The impact of batch retrievals on throughput performance of a carousel system serviced by a storage and retrieval machine. *International J. Production Economics*.

JIJU A. 2014. *Design of Experiments for Engineers and Scientists*.

JOO AE LEE, YOON SEOK CHANG, HYUN-JIN SHIM & CHO, S.-J. 2015. A Study on the picking process time. *Procedia Manufacturing*, 731-738.

JOSIP HABAZIN, ANTONIA GLASNOVIC & BAJOR, I. 2017. Order Picking Process in Warehouse: Case study of dairy industry in Croatia. *Traffic&Transportation,* 29, 57-65.

KAVEH AZADEH, RENÉ DE KOSTER & ROY, F. 2019. Robotized and Automated Warehouse Systems: Review and recent Developments. *Transportation Science,* 53(4), 917-945.

KRIEHN T. & FITTINGHOFF M. 2018. Impact of class-based storage, sequencing of retrieval requests and warehouse reorganisation on throughput of shuttle-based storage and retrieval systems. *FME Transactions,* 46, 320-329.

LERHER T. 2017. Design of Experiments for Identifying the Throughput Performance of Shuttle-Based Storage and Retrieval Systems. *Procedia Engineering,* 187, 324-334.

LERHER T., EDL M. & ROSI B. 2014. Energy efficiency model for the mini-load automated storage and retrieval systems. *nt J Adv Manuf Technol,* 70, 97-115.

LERHER T., EKREN B. Y., SARI Z. & ROSI B. 2015. Simulation Analysis of Shuttle Based storage and Retrieval Systems. *Tehnicki vjesnik/Technical Gazett,* 23(3), 715-723.

LERHER T., EKREN B.Y., SARI Z. & B, R. 2016. Method for evaluating the throughput performance of shuttle based storage and retrieval systems. *Tehnicki vjesnik/Technical Gazette* 23(3):715–723.

LERHER T., FICKO M. & PALCIC I. 2020. Throughput performance analysis of Automated Vehicle Storage and Retrieval systems with multiple-tier shuttle vehicles. *Applied Mathematical Modelling,* 91, 1004-1022.

LORENC A. & LERHER T. 2019. Effectiveness of product storage policy according to classification criteria and warehouse size. *FME Transactions,* 47, 142-150.

M.LAW, A. 2015. *Simulation Modeling and Analysis,* New York, McGraw-Hill Education.

MALMBORG CHARLES J. 2002. Conceptualizing Tools for Autonomous Vehicle Storage and Retrieval Systems. *International Journal of Production Research,* 40:8, 1807-1822.

MARCHET G., PERROTTI S., MELACINI M & TAPPIA E. 2013. Development of a framework for the design of autonomous vehicle storage and retrieval systems. *International Journal of Production Research,* 51:14, 4365-4387.

NOBUTAKA KIMURA, KIYOTO ITO, TAIKI FUJI, KEISUKE FUJIMOTO, KANAKO ESAKI, FUMIKO BENIYAMA & MORIYA, T. 2015. Mobile Dual-Arm Robot for Automated Order Picking System in Warehouse Containing Various Kinds of Products. *International Symposium on System Integration (SII)*, 332-338.

ONLINE. 2021. *Apriori Algorithm in Data Mining: Implementation with Examples* [Online]. Available: https://www.softwaretestinghelp.com/apriori-algorithm/ [Accessed 02/04/2021].

PACHEMSKA T.A., LAPEVSKI M. & R., T. 2014. ANALYTICAL HIERARCHICAL PROCESS (AHP) METHOD, APPLICATION IN THE PROCESS OF SELECTION AND EVALUATION *International scientific conference*, 373-380.

RICHARD BORMANN, BRUNO FERREIRA DE BRITO, JOCHEN LINDERMAYR, MARCO OMAINSKA & PATEL, M. 2019. Towards Automated Order Picking Robots for Warehouses and Retail. *Computer Vision Systems.*

RIFKI FAHRIAL ZAINAL & FARDANTO SETYATAMA 2016. ITEM ARRANGEMENT PATTERN IN WAREHOUSE USING APRIORI ALGORITHM (GIANT KAPASAN CASE STUDY). *Journal of Electrical Engineering and Computer Sciences,* 1.

ROSI B, GRASIC L, DUKIC G, OPETUK T & LERHER T 2016. Simulation based performance analysis of automated single-tray vertical lift module. *International J. Simulation Model,* 15**,** 97-108.

SHASHANK KUMAR, BALKRISHNA E. NARKHEDE & JAIN, K. 2021. Revisiting the warehouse research through an evolutionary lens: a review from 1990 to 2019. *International Journal of Production Research*.

VASILI M.R., TANG S.H. & M., V. 2012. Automated Storage and Retrieval Systems: A Review on Travel Time Models and Control Policies. In: Manzini R. (eds) Warehousing in the Global Supply Chain. *In:* SPRINGER, L. (ed.).

VISHWESH SINGBAL & K.ADIL, G. 2019. A Simulation Analysis of impact of design and storage policy on performance of single-crane multi-aisle AS/RS. *Science Direct,* 52-13**,** 1620-1625.

YASMEEN JAGHBEER, ROBIN HANSON & JOHANSSON, M. I. 2020. Automated order picking systems and the links between design and performance: a systematic literature review. *International Journal of Production Research*.

YUGANG YU, RENÉ B.M. DE KOSTER & GUO, X. 2015. Class-Based Storage with a finite number of items. *Production and Operations Management,* 24**,** 1235-1247.

ZAERPOUR N., DE KOSTER RENÉ B. M. & YU, Y. 2013. Storage policies and optimal shape of a storage system. *International Journal of Production Research,* 51**,** 6891-6899.