Delft University of Technology

Master Thesis

Transport Infrastructure and Logistics

Maximizing the output of a fully automated warehouse

Improving the output performance of the Optilogx box by scheduling

by

Pim Geurts

1268163

Report number: 2012.TIL.7814

December, 2013

Preface

The research described in this report has been done commissioned by ORTEC Zoetermeer and is used as thesis project to complete the master Transport Infrastructure and Logistics at the TU Delft. This report describes the development of a tool used for the scheduling of outfeed actions in a fully automated warehouse.

Special thanks goes to my graduation committee for their supervision and guidance during this thesis project. First of all, I would like to thank professor G. Lodewijks for your guidance and feedback. Secondly J.C. van Ham for your interesting pointers and our discussions during my thesis work. I would like to thank M. Duinkerken for your input in the last weeks and being part of the committee. Finally I would to thank M. Martens for making it possible to do my theses at ORTEC.

This thesis is the final step of my study at the TU Delft. I would like to thank my family and friends for all their support during the years at the TU Delft.

Pim Geurts
Delft, the Netherlands
December 2013

Author:	Pim Geurts	
Student ID:	1268163	
Email:	p.c.geurts@student.tudelft.nl	

Committee:	*Chairman:*	Prof.dr.ir. G. Lodewijks
	Internal Supervisor:	Prof.dr.ir. G. Lodewijks
	Internal Supervisor:	dr. J.C. van Ham
	Internal Supervisor:	ir. M.B. Duinkerken
	External supervisor ORTEC:	ir. M. Martens

Company: ORTEC, department ORTEC Business Development

Summary

Problem and objective of the study

Most warehouses currently used are still designed as traditional fishbone layouts. Pallets are transported and shelved by a pallet truck driver. This time-consuming process repeats itself when pallets are retrieved from the warehouse. In order to make this process faster and less labor-intensive, fully automated warehouses are developed. Since a couple of years ORTEC is involved in the implementation and development of software for the control of fully automated warehouses. One of the projects they currently work on is the cold-store warehouse at Hellendoorn in the Netherlands. The mechanical system involved is called the Optilogx box, which has an operating temperature of almost -30° Celsius. The main advantages of the Optilogx box are the possibilities to store more pallets per volume, use far less personnel than in a conventional warehouse and the possibility of achieving high energy savings.

From the outfeed of the warehouse in Hellendoorn, the pallets will be picked up by pallet-truck drivers and loaded into a truck. The requests for the pallets are done in orderbatches which contain between the 26 and 33 pallets. In some situations the average time it takes to transport a pallet to the outfeed is way longer than the technical capacity of the system prescribes. Delay or time-lag arises inside the system, resulting in long outfeed times. These long outfeed times have a direct influence on the loading times of the trucks at the outfeed.

Especially in the case of a relatively full warehouse, the time-lag in the system is unacceptable. Therefore the goal of the presented study is to determine what is causing the time-lag inside the warehouse and to find a way to bring this time-lag to a minimum. The Sashimi model is used as a methodology for this study, with the advantage of feedback between the different phases of the research.

Analysis

An extensive analysis is performed in which the following aspects are discussed: First a small market research is done in the field of (automated) warehouses. Secondly a description of the Optilogx box is presented in which the working principle of the box and the software lay-out developed by ORTEC is described. The top layer of the ORTEC software (WPMS) is responsible for the selection of requested pallets of an outfeed orderbatch, and the generation of the start and end position of every pallet involved in this outfeed process. The main cause for the time-lag inside the system is the fact that WPMS does not take time or resources into account when generating the pallet actions. On the other hand the necessity to dig-out pallets does contribute to the amount of time-lag on many occasions. These dig-out actions are necessary in an unsorted

warehouse, or in the situation in which the amount of different products outnumbers the lanes inside the warehouse. From the analysis the conclusion is drawn that the Optilogx box has many advantages in comparison with other (automated) warehousing systems. However, the software can be improved to enhance the performance of the system. In the current situation, actions are performed in the same sequence as they are generated by the software. A model is developed to minimize the time-lag, by scheduling the outfeed actions generated by WPMS.

Design

The situation of the outfeed at the Hellendoorn warehouse shows resemblance with a jobshop and flowshop problem. The pallet movements in the Optilogx box will make use of a selection of machines; not every pallet moved to the outfeed uses every machine. Therefore the problem is not similar to a standard or permutation flowshop problem, but also shows some aspects of a jobshop problem without pre-emption. This combination of a job- and flowshop problem has no standard solution algorithm available for solving this problem. Therefore a Mixed Integer Programming model is constructed. The objective function of the model is to minimize the variable of the total timespan, while taking the values of the parameters and sets as starting point and satisfying the conditions of the constraints. The goal of the model therefore is described as: optimizing the sequence in which the pallets will be moved to the outfeed, with the goal of scheduling the last action as early as possible in the time and therefore minimizing the total timespan. The scheduler is developed in the modelling environment AIMMS and uses the CPLEX 12.5 solver.

The infeed for the model are the actions involved in the outfeed process. These jobs are combined with a routing table in which the processtime per resource is present. The MIP model will pick up these jobs and the other data, to reschedule them into an optimized sequence. This sequence is then sent to the executive layer of the software (CP), which ensures the actions are carried out by the system in the new sequence. Because of the limited available time, it is not possible to build the scheduler into the software of ORTEC. A recommendation is done for the place to implement the scheduler into the existing OWCO software.

The scheduler was developed as a standalone tool and is specifically configured to work at the outfeed procedure of the Optilogx box found at the Hellendoorn warehouse. The generic design however will make it possible for the model to work for an Optilogx box with a different configuration. For example a different amount of Vertical or Horizontal Transporters inside the Optilogx box. This new configuration can be implemented in the model with the necessity of only minor adjustments. The model can also be used for the outfeed procedures of an Optilogx-satellite warehouse, again with only minor adjustments in the configuration.

Model results and performance

The emulation tool is used to test the performance of the model. This emulation software is already developed by ORTEC, and makes it possible to emulate pallet movements inside the

warehouse. With this tool, test scenarios are run to test the performance of the scheduler. First a total of 11 cases are run representing different scenarios inside the Optilogx box. The model presents the optimal sequence in every single case. There is however a problem; the calculation times used in these cases are unacceptable for the use in the current warehouse. An orderbatch will be requested at the moment the truck is ready at the loading dock, therefore there is no time available for the calculation of the optimal sequence in the scheduler. The time used by the scheduler will result in direct waiting time for the pallet-truck driver at the outfeed. To solve this time problem, the scheduler was restricted to a maximum calculation time of 10 seconds. The same test cases are put into the model, to determine the differences in performance with and without the time restriction.

The outcomes of both these tests are promising. In the situation of no calculation time restriction the model optimizes the sequence, meaning no more time-lag is present in these outfeed procedures. The big disadvantage is the amount of calculation time needed in some situations. The second scenario will not fully optimize the outfeed sequence in every case, but will still reduce the amount of time-lag from an average test scenario of 21.7% time-lag in the current situation to an average of only 1.1% time-lag at the same test scenarios. In the real warehouse this will lead to an estimated timespan reduction between the 10% and 15% dependent on the capacity utilization. The reduction in outfeed times have a direct effect on the loading times of the trucks. Therefore the trucks will also be loaded 10% to 15% faster, leading to less waiting time for the trucks and fewer additional expenses.

Recommendations

The test results can lead to a recommendation towards ORTEC, implying on the possibility to reduce a high amount of time-lag by implementing the scheduler. The recommended place for implementation of the scheduler will be inside the top layer of software (WPMS) in the OWCO software.

In short the recommendations following from this report are:

- Implement the scheduler inside WPMS
- Use the scheduler at the Hellendoorn warehouse for scheduling the job sequence
- Tweak the solver for a possible calculation time decrease
- Do a lower bound calculation and let the solver start at this lower bound to reduce calculation time
- Make sure the warehouse is sorted by improving the Housekeeping function
- Replace the outfeed MUP by simple chain conveyer lanes, to increase the outfeed capacity of the warehouse
- Further research must be done to determine the amount of calculation time needed by the scheduler to come to a sufficient solution in a minimal amount of time.

- Further research must be done on the possibility to have the orderbatch requests prior to the moment the pallets are needed at the outfeed.

Key Words: automated warehousing, logistical model, outfeed, job shop, flow shop technical capacity, scheduling.

Abbreviations

3PL	-	Third party Logistics
AGV	-	Automated Guided Vehicle
CP	-	‘Connactive Processes’
DB	-	Database
ERP	-	Enterprise Resource Planning
FIFO	-	First In First Out
FMS	-	Flexible Manufacturing system
HT	-	Horizontal Transporter
LIFO	-	Last In First Out
MUP	-	‘Multifunctioneel uitslag punt’
NP-hard	-	Non-deterministic Polynomial-time-Hard
OWCO	-	ORTEC Warehouse Control and Optimization
PLC	-	Programmable Logic controller
VRP	-	Vehicle Routing Problem
VT	-	Vertical Transporter
WPMS	-	Warehouse Planning & Management System
XML	-	Extensible Markup Language

Contents

1 Introduction

This chapter provides the introduction to this research project, by firstly giving the current situation on the subject of automated warehouses and the different projects that ORTEC is currently working on. Next, the possible improvements of the system developed by ORTEC will be discussed, which leads to the problem description. This problem description will be followed by the methodology used throughout this report. Finally this chapter closes with the report outline.

1.1 Current situation

Currently, most warehouses are still designed as traditional fishbone layouts. Every Pallet that comes in is transported to a place in the warehouse by making use of a pallet-truck and its driver. For example in a production plant: a new pallet arrives at the outfeed of the production plant and then continuous to the infeed of the attached warehouse. From there the product is identified (in most of the cases by bar-code scanners) and a place in the warehouse is assigned to the pallet. Then the warehouse personnel transport the pallet to its place by making use of e.g. a pallet-truck. This time-consuming process continues as long as the production plant is producing. If a customer or a distribution center places an order for a certain amount of pallets, this order will arrive at the warehouse and the right pallets are selected. The pallet-truck driver then needs to locate the right pallets and transport them to the outfeed. Again, this is a time-consuming process for both personnel at the outfeed of the warehouse, as well as the third party logistics provider's (3PL) truck driver whose truck is loaded with products for the customer. In order to make this process faster and less labor-intensive for both pallet storage and the loading of trucks, fully automated warehouses are developed. More and more companies are considering the option of using a fully automated warehouse. This concept of automation is not an entirely new one: in 1969 Philips started looking at the feasibility of automated storage warehouses (Tulp, 1974), and started with the built a few years later. Also Centraal Boekhuis has been making use of a version of an automated system for a few decades now (Redwood business solutions, 2008). While this is not a new concept, the market for these fully automated warehouses is still growing. This leads to new companies interested in providing one or more services necessary for the implementation of these automated warehouses. ORTEC is one of these relatively new companies interested in the market of fully automated warehouses.

Right now, the warehousing department of ORTEC is busy with the implementation and development of new software for the control of fully automated warehouses. The projects they are currently working on are an automated warehouse for a big and famous ice cream production plant at Hellendoorn in the Netherlands and an automated warehouse used by a big ice cream production plant in the UK near Liverpool (FreezeServe). These warehouses are both live and running, but the projects are not finalized yet. Still some improvements have to be made to make the warehouses run even better than they currently do. An important resemblance between these two warehouses is: they are both directly linked to a production plant.

ORTEC's future project on automated warehousing will be to deliver the software for a big production plant in France. This warehouse will be used by a big food company in France. The similarity with ORTEC's current live warehouses is that this warehouse is also directly linked to the production plant of the food company in France. These warehouses are therefore all positioned at the same location as the production plant. This is not the case for every single automated warehouse ever built is, but for ORTEC this is what they are specializing in right now. Another point of interest is the fact that these warehouses handle either cooled (warehouse in France) or frozen products (Rumst, Hellendoorn and FreezeServe). This is not a coincidence, but can be explained by the fact that these frozen warehouses are more expensive to run. Not only the electrical costs for keeping the temperature low are involved, but also the personnel costs. In these warehouses which have an operating temperature of almost -30° Celsius, personnel can only be operational for a short amount of time because of health and safety regulations. This results in the situation in which such a manually controlled warehouse is very expensive and labor-intensive to run. Therefore the option to fully automate a warehouse operating at low temperatures can be a reasonable and cost-efficient solution, even when taking the high investment costs of an automated warehouse into account.

The main advantages of these automated warehouses are the possibility to store more pallets per volume and to use less personnel than in a conventional warehouse. Also the pallet-registration and the inventory of the warehouses are easily accessible. Another advantage worth mentioning is the management of 'best-used-before' dates and restrictions on the pallets, which will be far easier to carry out if a warehouse is fully automated.

1.2 Improvements

As mentioned before, the current state of ORTEC's automated warehousing projects, is 2 out of the 3 projects they currently work on are up and running. These projects are in the state of finalization, which means that the warehouse is working, but the software is still being optimized in order to provide the customer with an even better/faster working warehouse. The third project is still in the design phase and the build will be started next year.

The automated system in the 2 'live' projects, is able to take-in the new pallets at the infeed and store them for a certain amount of time. Finally the system will transport the requested pallets to the outfeed when a new order for a pallet arrives. From the outfeed, the pallet is picked up by a pallet-truck driver and transported to the 3PLs truck. When the total number of pallets currently stored in the warehouses is low (for example a capacity utilization less than 25%), than in most cases the pallets will come out relatively quick mainly because the requested pallets are easily accessible. There is always a bottleneck that determines the maximum handling speed of the outfeed procedure of pallets. In the ideal situation, this bottleneck will be the pallet-truck driver at the outfeed who transports the pallets from the outfeed to the 3PL's truck. The automated warehouse will in that case get the pallets faster to the outfeed than the pallet-truck driver(s) can handle. This situation does not occur very often in the implemented warehouses. Therefore there are still possibilities for improvement, especially in situations where the warehouse has a capacity utilization of more than 75% of the total storage capacity. In this situation of a relatively full warehouse, the average time it takes to transport a pallet to the outfeed will go up really fast. This is a main problem in the automation of the warehouses, because it is in conflict with one of the big advantages of the automated warehouses; to be able to store more pallets per cubic meter, while keeping the outfeed of pallets fast. The warehouse is still able to store more pallets per volume, but if the outfeed is slow, this extra storage capacity is still useless. Therefore the bottleneck of the outfeed-speed (the amount of pallets per time-equivalent) is an important indicator to see how well the system actually works in the different storage utilization rates. In a situation in which the outfeed of the warehouse is the bottleneck instead of the truck loaders, we can define the time that the system is delayed as the *time-lag* inside the system. In chapter 2.4 this will be further discussed. The time-lag will be 0 in the ideal situation, but in the real warehousing situation it is tried to keep the time-lag as small as possible. The smaller the better, because every extra second it takes to move the pallets to the outfeed, is extra waiting time for the 3PL's truck driver and thus fewer trucks can be loaded per day.

In order to make the warehouse as fast as possible, ORTEC makes use of a mechanical system called the Optilogx box[1]. This system is designed and patented by Nico van Rijn in cooperation with SPIE Building-Systems[2]. This Optilogx box consists of a number of buffer lanes in which a combination of pallet racks and chain conveyors is constructed. This means the buffer lanes themselves are capable of moving the pallets through the lanes. The pallets in the different buffer lanes are presented in figure 1.1 with variable colors. Two other main parts inside the Optilogx box used to move the pallets, are the shuttles which transport the pallets in the horizontal plane (the brown devices in front of the buffer lanes), and the vertical transporter which is used for the vertical transportation of the pallets. A more detailed description of the Optilogx box will be given in chapter 2.3.

[1] http://www.spie-nl.com

[2] http://www.spie-nl.com/en/divisions/spie-industry/products-and-services/optilogx.html

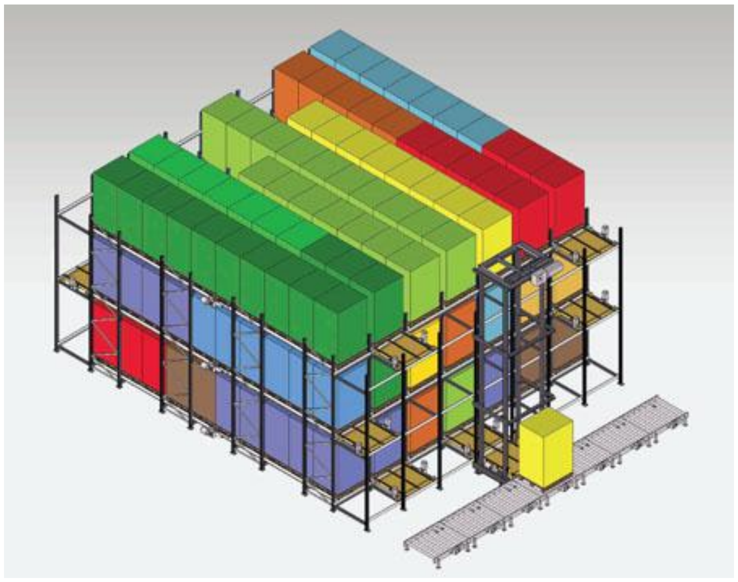


Figure 1.1: The Optilogx box, designed and patented by Nico van Rijn [1]

With this Optilogx box inside the warehouse, the automated system manages to store and reorganize the pallets before moving the selected pallets to the outfeed. To store every pallet and move the correct pallets to the outfeed, ORTEC makes use of a software component which is called the 'Optimizer' of the software. This optimizer is capable of selecting a pallet place for the incoming pallets inside the warehouse and selecting the correct pallets when a new outfeed order arrives. It will also give the start and end position of every pallet involved in the outfeed order. This optimizer is the top layer of the software and can be seen as the 'brain' of the warehouse control software. As said before, the average time it takes to transport a pallet to the outfeed will go up really fast if the warehouse gets fuller. Therefore there is still room for improvement, especially when focusing on this top layer of the software.

1.2.1 Minimization of the time-lag

Automated warehouses that run with the ORTEC OWCO software are still producing an amount of time-lag dependent on the storage situation inside the warehouse. Especially in the case of a relatively full warehouse, the time-lag caused by the system is unacceptable. Therefore the goal of this research is to determine what is causing the time-lag inside the warehouse. The aim of this thesis is to find a way to bring this time-lag to a minimum.

While the total objective of this research is to improve the performance of the system, not all aspects involved in automated warehousing will be discussed in detail. The focus of this research will be on the minimization of the lag-time inside the Optilogx box. Because the Optilogx box can be seen as the heart of the system (every in and outgoing pallet will go through the Optilogx box), minimizing the time-lag leads to an improvement of the entire system. In order to achieve this total performance improvement, firstly an analysis must be done to give a description of the current situation and find the problems and opportunities. This analysis phase and the total methodology will be discussed in the next paragraph.

1.3 Methodology

In order to find an improvement in the reduction of the time-lag inside the system, a methodology will be used. From literature some common methodologies have been found (OSQA, 2009) all with different approaches and therefore useable for different kind of problems. The focus of this project is on the development of a model that can reduce the time-lag originated in the Optilogx box. Because the OWCO system is entirely software driven, the newly designed model should be integrated into the existing software. Therefore some standard software design methodologies will be discussed. These are the waterfall models, commonly used in the development of new software.

1.3.1 Waterfall Models

A model often used in the development of computer software, is the waterfall model. This model is a sequential design process consisting of several phases in which the total progress can be seen as a steady flow downwards (a waterfall). These phases are given in figure 1.2 and consist of the conception, the initiation, the analysis, design, construction, testing and finally the maintenance of the software. The main characteristic of this system is that every phase must be fully completed before the next phase can begin. The name 'waterfall model' comes from the progress flow that flows from top to bottom like a cascading waterfall. This is just a schematic representation of the model, meaning the length of the different phases in the time do not represent the amount of time actually spent on each of these phases.

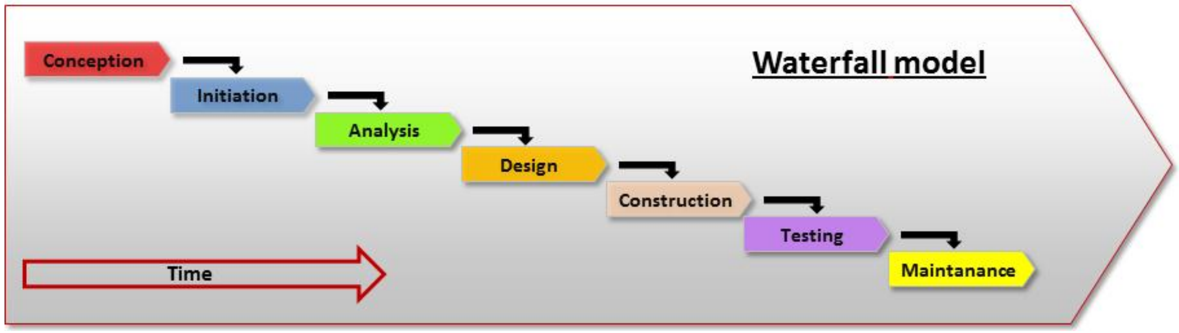


Figure 1.2: The waterfall model

A derived method from this waterfall design approach is the Sashimi model developed by Peter Degrace (Matković, 2010). The big difference with the standard waterfall model described above is the sequence in which the phases can be performed. While in the waterfall model every phase has to be completed in order to start a new phase, in the Sashimi model the next phase can already start while the current phase is not finished yet. Figure 1.3 shows the phases of this model which have a small overlay. Again this is just a schematic overview of the model, so the overlay of the different phases does not represent the actual proportion of the overlay between the phases. (Matković, 2010). The black arrows represent the normal flow from phase to phase, while the red arrows represent the interchange between the different phases. First the red open arrows, for example from the analysis phase to the design phase (while the analysis phase is not finished yet). Secondly the smaller red filled arrows, for example from the design phase back to the analysis phase. This interchange between the different phases can have the advantage of more interaction between the different phases. This means the design of the model can already start without the necessity of finishing the analysis phase, but there will also be the possibility to fall back on the analysis phase when designing the model.

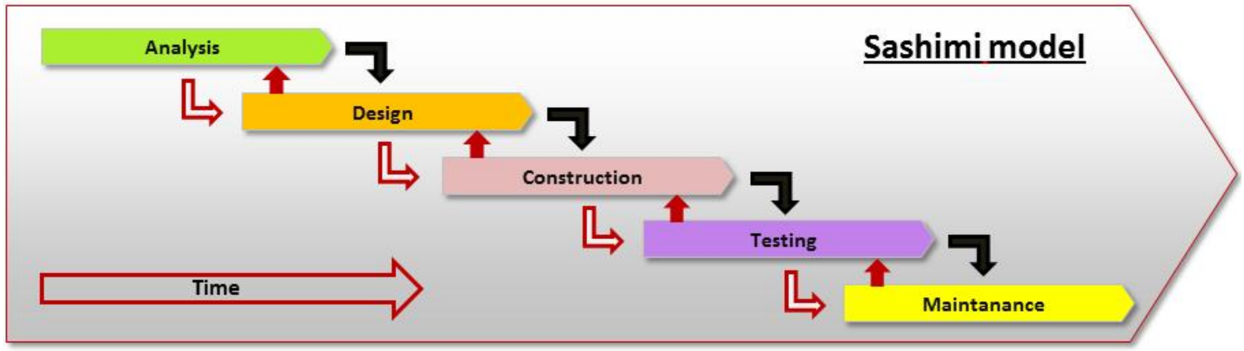


Figure 1.3: The Sashimi waterfall model

1.3.2 Project design approach

There are two very important aspects when selecting an appropriate design model; the overall success in practice of the model and the familiarity of the user with the model. These two criteria led to the design approach based on the sashimi model. This design model is given in figure 1.4.

The difference with a standard design project, is the fact that (a part of) the software is already developed and running at the customer's warehouses. ORTEC has been working on this software for a few years now, but the implementation of the software gave the results that neither the customer nor ORTEC was fully pleased with. The basic functions of the software work, but there are still some possibilities for improvement. Therefore this research will focus on the improvement and the development of a (new) component in the software.

In order to give a clear overview of the current situation, the problems/opportunities and the already developed software, this research project will start with an extensive analysis phase. This analysis phase will be performed, with the objective of analyzing the current situation and determine the problems inside the OWCO automated warehouses. This analysis will also include a description of the current situation of the system, and a small analysis of the current warehousing market and the comparable systems in this market. In order to achieve the main goal of determining the problems inside the system and to form the main research question, 3 analysis-questions have been developed:

1. ***What are the characteristics of a fully automated warehouse in comparison with a conventional warehouse? And are there any examples of comparable fully automated warehousing systems that are already in use?***

2. ***What is the technical capacity of an automated warehouse, and which factors influence this technical capacity?***

3. ***How does the optimizer manage the Optilogx box in the current situation? And what are the main bottlenecks involved causing the time-lag in the Optilogx box?***

The first question includes the literature research on existing warehousing systems and the current (automated) warehousing market. The second question will provide information about the technical capacity and the influencing factors. The third question of the analysis phase will give a description of the optimizer used in the OWCO software to control the warehouse. This analysis phase will conclude with the presentation of the research question and sub-questions constructed to guide towards the development of a new model capable of reducing the time-lag in the Optilogx box.

The second part of the project will be the design phase. For this phase the Sashimi model can be used as an example, because this represents the model with the best practical means in relation to software development. This second part of the report will contain the design, the construction, the validation and verification, and finally the testing of the new model in order to provide an answer to the research question stated in paragraph 2.8. This phase will provide an answer to the research question, by making use of the sub-research questions. These answers will be reported and the different steps taken in order to come to the developed model will be explained. After the development, the model has to be validated and verified. This will be done according to the paper on model validation and verification by R.G. Sargent (2004), and is also included in this design phase of the report. The final step is to test the model, which will be done by making use of emulation software available for the Optilogx box. This software is already developed by ORTEC, and makes it possible to simulate pallets and pallet movements inside a warehouse. This allows the tester to simulate the real movement of pallets in the warehouse (including in- and outfeed) and create all different kind of testing scenarios. The normal testing scenario at the warehousing department of ORTEC is to first ‘positively test’ if the new functionality is working in the software (done by the software developer himself). This means the new functionality is written in the software and it is tested for one or more specific cases to see if the software performs this new functionality as it should. When everything works as it should, the next phase of testing will be carried out; this is called the ‘negative testing’ of the software. The tester (this does not have to be the same person as the software developer) tries to come up with all kind of different scenarios to see if it can crash the software, or simulate a situation which the software cannot handle. These problems in the software or so called ‘bugs’ will then be reported and solved by the software developer. Then the negative testing procedure will be carried out again, till the software is working as it should. This entire testing procedure will be done to see if the developed model works, and therefore the testing has to be completed before the performance is compared with the older versions of the warehousing software.

The last phase in the Sashimi model is the maintenance of the design. This last phase is outside the scope of the deliverables for this report, and therefore will not be discussed extensively in this report. However the report does suggest recommendations for either maintenance or improvements on this system in the future.

The total approach is shown in figure 1.4 with the different steps in this methodology. Also the subdivision into the two different phases can clearly be seen in this figure: on top the analysis phase with the current situation and the problem definition followed by the design phase with the design, construction, validation and verification and the testing of the model. Outside these 2 phases, the start of the project is given by the client statement. In this specific case the improvements of the OWCO software (ORTEC Warehouse Control and Optimization) can be seen as the client statement given by ORTEC. The link between the two phases is given by the blue line on the right of the figure. This feedback loop represents the comparison between the problem definition in the analysis phase and the test results of the software in the design phase. After the testing, the conclusions will be drawn and the report is constructed. This is indicated in the figure as the last blue balloon: the documentation (Report and Recommendations).

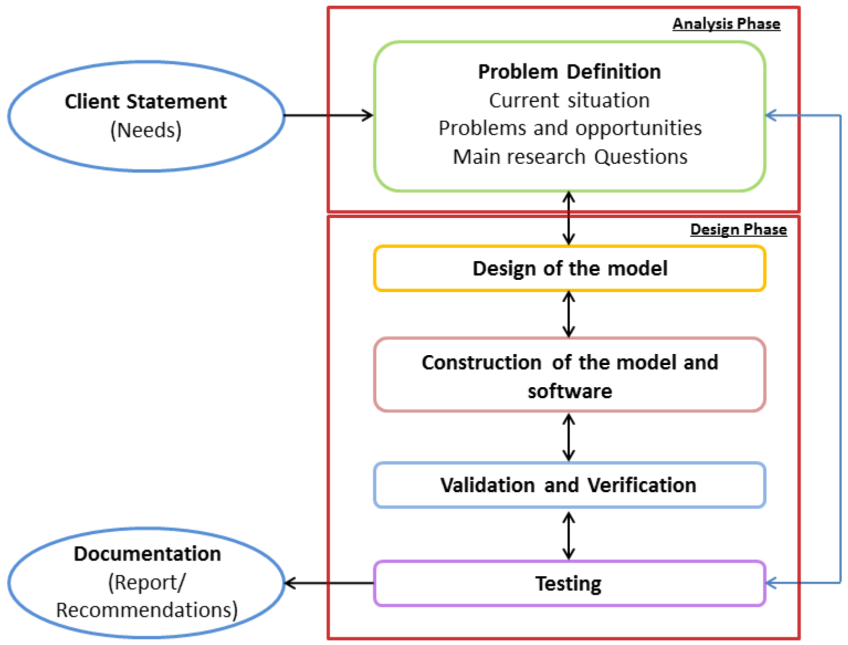


Figure 1.4: Project methodology

1.4 Report Outline

The previous paragraphs give a small introduction, including the description of the current warehousing situation and the current status of the warehousing projects in which ORTEC is involved. The possible opportunities for improvement are shortly discussed and the methodology used throughout this entire report is presented. Some more background information can be found in appendix A.

Chapter 2 will start with the analysis phase in which the questions described in paragraph 1.3 will be analyzed and answered. A small market research will be done in the field of (automated) warehouses, and an extensive description of the Optilogx box will be given. This chapter will also contain a description of the technical capacity of the Optilogx box, the working principle of the system and the communication between the different layers of software inside the box. Finally the analysis phase will be concluded and the research question will be presented.

The next chapter will start with the description of the different models that are comparable with the situation inside the Optilogx box. A software choice is presented for the development of a new model capable of solving the problem inside the Optilogx box. The link will be made between the developed model and the real-case scenario inside the Optilogx box at Hellendoorn. This chapter continuous with the presentation of a solver, which is used inside the model. Then the design of the model and the implementation of the model will be discussed along with the mathematical formulation of the model. Finally the model verification and validation will be discussed.

In chapter 4 the model results and performance will be discussed. Several cases will be presented in order to determine the results generated by the model and the performance of the model. For the testing of these cases an emulation tool of ORTEC will be used in order to simulate the pallet movements inside the warehouse of Hellendoorn. This chapter will end with the conclusions drawn from the cases.

The final chapter will focus on summing up the conclusions which can be made throughout the report. Also the recommendation towards ORTEC will be discussed.

2 Analysis

This chapter consists of the analysis phase of the research, in which the three questions constructed in paragraph 1.3 will be answered. Paragraph 2.1 will focus on the difference between automated and non-automated warehouses. In combination with paragraph 2.2, which describes other automated warehousing systems; these first two paragraphs will answer the first question. Paragraph 2.3 will give a description of the Optilogx box with all its (in- outfeed) components. The second question will be answered in the paragraph 2.4 in which the technical capacity will be elaborated on. The two paragraphs 2.5 and 2.6 will provide the working principle of the system. In paragraph 2.5 the third question will be answered and the communication between the different programs will be explained, while the sixth paragraph describes the working principle of the Optilogx box. Finally the chapter is closed with the conclusion of the analysis phase and the research question for the design phase will be presented with the sub-questions. The next chapter will focus on answering this research and sub-questions.

2.1 Differences automated versus non-automated warehouses

There are different kinds of warehouses available for all different kind of functions. The last decade's lots of improvements have been made on warehouses, resulting in a variety of warehouse types able to fulfill various functions. The classification of the different warehouses can be made from the 3 following viewpoints: On the basis of ownership, the service rendered and on the basis of structure. On the basis of ownership is again divided into sub-categories, in which the private and the public warehouse are the main groups. Also the viewpoint on the basis of the service rendered consists of some sub-categories. The main groups in this sub-category are: The specific commodity warehouse which has the functionality to store specific goods like cotton or wool, and the general merchandise warehouse. In this warehouse all different kind of product can be stored that don't need any specific storage facility. The last group in this sub-category is the cold storage warehouse. In this warehouse mostly perishable goods are stored. The warehouses can also be divided on the basis of structure, including the conventional warehouse (a big hall for the storage of goods) as well as portable warehouses. Also silos and bins can be classified in this category.

Not all existing types of warehouses will be discussed in this paragraph, but only the types that are closely related to this research project. This research project focuses on the automated version of a conventional warehouse with storage racks for pallets, also the various layouts in the warehouses will be discussed. Therefore the differences in warehouses based on ownership will be neglected. Also the specific commodity of the cold storage plays a role. The warehouses involved in this research are all cold warehouses that contain perishable goods. In the specific case of a cold storage warehouse, it is even more important to store the pallets as closely together as possible, because every cubic meter that does not have to be cooled in the warehouse saves energy and therefore money.

The different layout types will be compared according to the strong and weak points of every layout type. Another important criterion for comparison is the amount of pallets that can be stored inside the warehouse. This will be compared according to the number of pallets that can maximally be stored per volume of the warehouse. With the assumption that the space between the storage rack levels is the same for every different layout type, the comparison can be made with the number of pallets per surface area (pallets per m^2), as well as the number of pallets per unit of volume (pallets per m^3). In the next paragraph the comparison will be made for the five most relevant layout types for a conventional warehouse.

2.1.1 Conventional warehouses

Classical warehouse

The first type of warehouse that is still widely used is the classical warehouse with storage racks. As can be seen in figure 2.1[3], this warehouse is a simple, clear and organized way of storing pallets. The pallets can be picked up by a pallet-truck which is able to drive in the aisles between the storage racks. The big advantage of this kind of warehouse is the possibility for the pallet-truck driver to reach every single pallet without moving any other pallet. This high reachability however, comes with a big disadvantage: only a limited amount of pallets can be stored in these kind of warehouses. The reason for this limited amount of pallets inside the warehouse is due to the space needed for the aisles between the storage racks. This leads to an area inefficient warehouse in which the number of pallets that can be stored is very low per square meter. The option of a classical warehouse can be considered if the area usage is not of big importance, but the investment costs for the hardware inside the warehouse is.

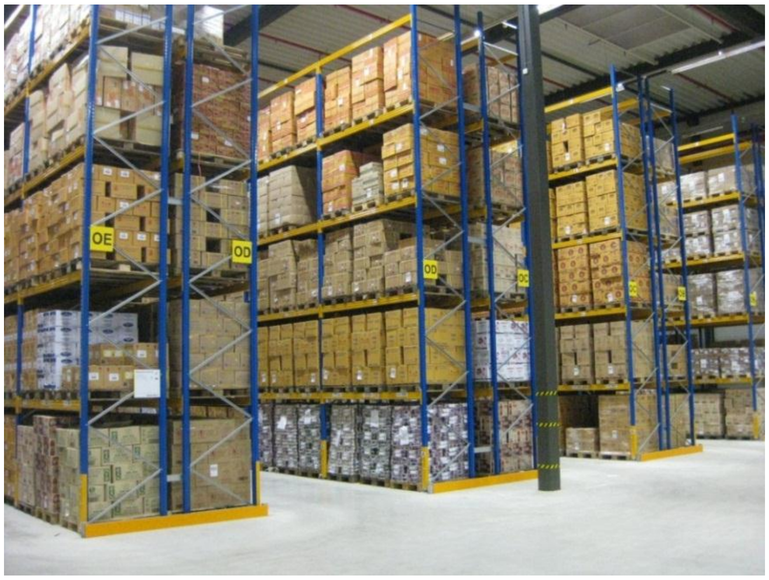


Figure 2.1: Classical warehouse [3]

[3] http://eyemerge.nl/verschillende-situaties

Narrow aisle racking

The second type of layout for conventional warehouses is the narrow aisle racking. This type of layout is very similar to the classical warehouse with storage racks. The main difference with the classical layout is that the storage racks are positioned closer together. This results in the possibility to store more pallets per square meter. Like in the classical layout, every pallet can be reached directly without the necessity to move any other pallet in the warehouse. There is however one big advantage for narrow aisle racking, the pallets cannot be retrieved from the racks with a conventional pallet-truck and therefore special pallet-trucks have to be purchased in order to be able to operate the warehouse. These warehouses are best used in configurations which need an efficient area usage[4], but still need access to every single pallet inside the warehouse. The figures show such a narrow aisle pallet truck combined with a narrow aisle storage rack.

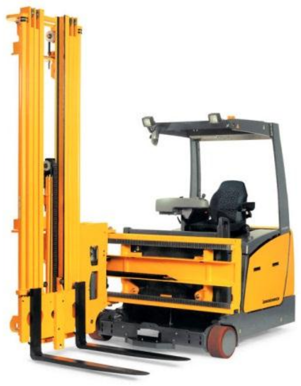


Figure 2.2: Narrow aisle pallet truck [4]

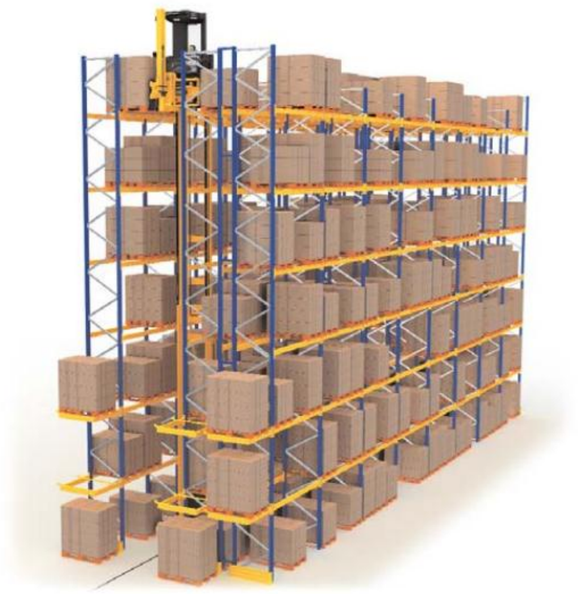


Figure 2.3: Narrow aisle racking [4]

Mobile racking with pallet truck

Another way to arrange the layout inside conventional warehouses is to make use of mobile racking. This type of layout can be seen in figure 2.5. The difference with the two layouts discussed above: these pallet racks can be moved in the horizontal plane. This means if a certain pallet is needed, the pallet racks will move in order to create an aisle for the pallet-truck. This can be compared with the mobile shelving system for file cabinets of for example Bruynzeel storage systems[5].

[4] http://www.directindustry.com/prod/jungheinrich-1078.html

[5] http://www.bruynzeel-storage.nl/Producten/Verrijdbare-archiefkasten/Compactus-Manual/

The advantage of this type of layout is that more pallets can be stored than in the two layouts above. Also every pallet that is requested can still be reached with the pallet-truck. The big disadvantage however is the movement of the storage racks in order to create an aisle at the right place. Not only will this cost a lot of energy (a large percentage of pallets will be moved in order to create an aisle for just one pallet) but this process also takes up some time. This means the pallet truck driver has to wait for the pallet racks to create the right aisle, before the driver can actually pick up the right pallet. Another problem is the disability of performing more than one action at the same time, because only one aisle can be created for a row of pallet racks. This has a negative effect on for example the in- and outfeed procedures, which than sometimes have to wait on each other. This means the mobile racking warehouse is preferably used in a configuration of products with a low throughput rate and a long storage time. This means the amount of movements per time-equivalent inside the warehouse is little.

Figure 2.4: Mobile shelving system [5]

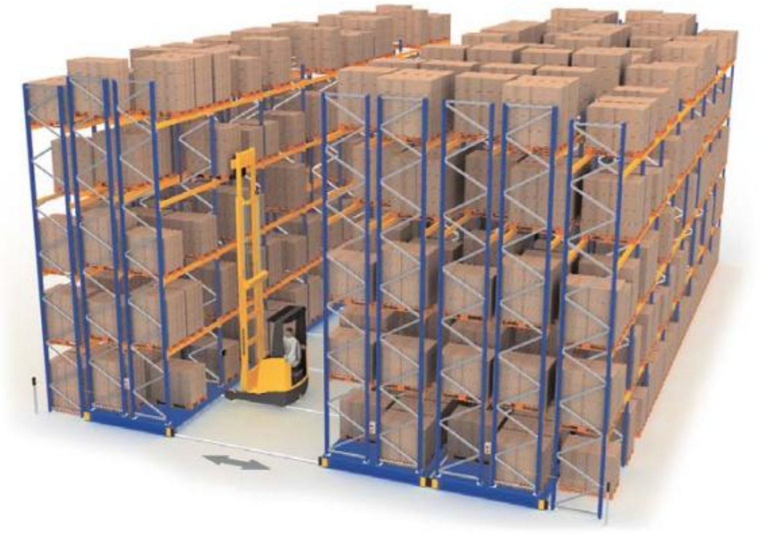


Figure 2.5: Mobile racking with pallet truck [4]

Push-back racking

Another way to store the pallets inside a warehouse is to make use of push-back racking. This system works with storage racks with conveyers or rollers that are positioned under an angle between 3% and 5% [4]. If a pallet is placed in a rack, gravity will always make the pallet slide down to the first position on that row. When a second pallet is placed in that same row, the first pallet will be pushed to the second place in that row. This way of storage can be really efficient when making use of the Last In First Out (LIFO) principle. Also the amount of pallets that are stored per area can be very efficient. However, there is a big dis-advantage: if for any reason there is a deviation from the LIFO principle, problems arise. Only the first pallet on a row can be reached and if the second pallet is requested, the pallet-truck driver first has to remove the first pallet. This will be a time consuming process that some warehouses just cannot afford. This kind of warehouse layout is preferably used in processes that are strictly kept to the LIFO principle.

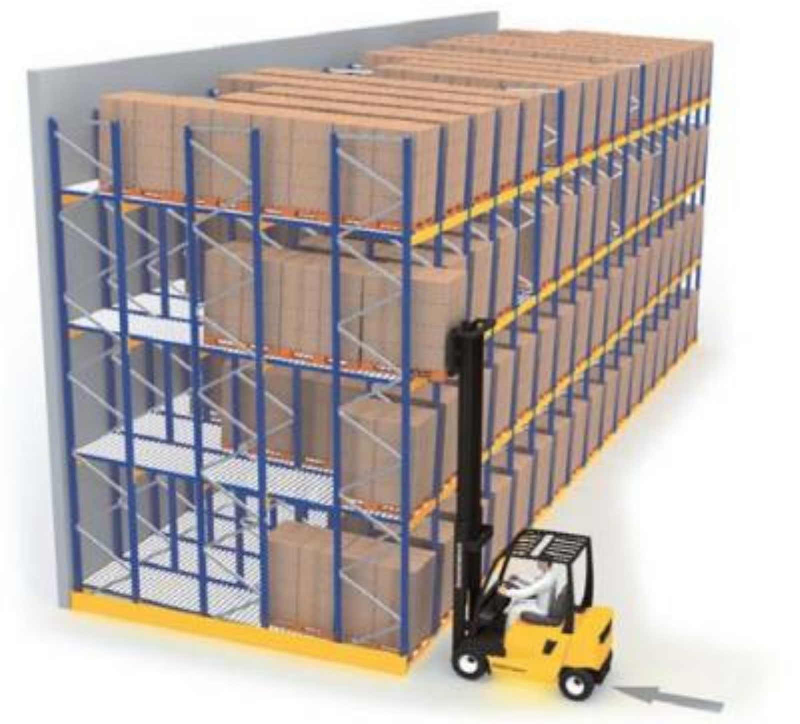


Figure 2.6: Push back racking [4]

Fully automated warehousing

In order to increase the amount of pallets stored per square meter of warehouse, the narrow aisle warehouse was developed. There are some automation variations possible for these kinds of warehouses, for example a narrow aisle automated warehouse, or a combination of a narrow aisle warehouse and a mobile racking warehouse. This automated combination can be seen in figure 2.7. This warehouse is fully automated and therefore does not make use of any pallet-trucks inside the warehouse. The big advantage is the possibility to store many pallets on a small amount of surface area[6]. But there is always a disadvantage: the system can only pick up one pallet at a time, so the in- and outfeed speed of the warehouse is limited. This system is really useful if only a small area is available and the in- and outfeed speed of the total system is not of big importance.

[6] http://www.daifukuchina.com

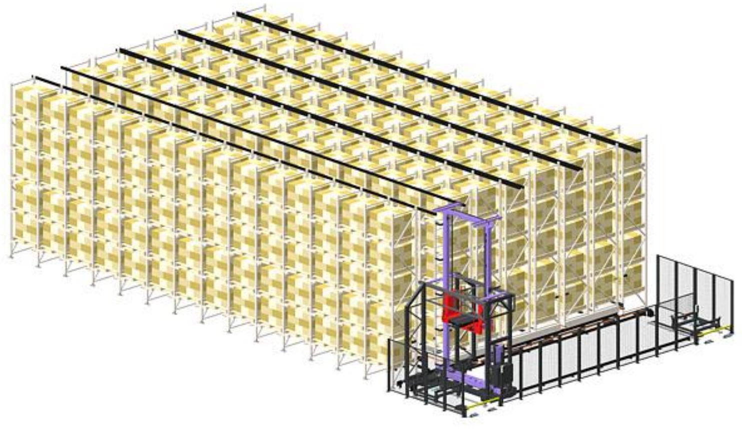


Figure 2.7: Automated combination warehouse [6]

In order to increase the amount of pallets that can be stored per square meter but also keep high in- and outfeed speeds possible, Nico van Rijn developed the Optilogx box. This fully automated warehousing system does not only have the advantage of storing pallets really close together, but also to sort and resort the pallets inside the Optilogx box. In figure 2.8 the fully automated Optilogx box can be seen on the right, in comparison with the classical layout of the warehouse on the left. As can be seen in the figure, the amount of pallets stored in the Optilogx box is far more than the amount of pallets stored in the classical warehouse layout. In this case: the number of pallets per level in the classical layout can reach a maximum of 296 pallets. In comparison, the number of pallets in the Optilogx box can reach a maximum of 663 pallets per level, more than twice as many pallets can be stored inside the Optilogx box.

Like for every warehouse-layout there are disadvantages. A big difference with the narrow aisle warehouse is the fact that there is no possibility in the automated warehouse to reach every single pallet. The pallets will be stored in lanes and can therefore only be retrieved if the pallet is moved to the front of a lane. Another main concern is, if for some reason the software or the mechanical hardware of the system fails, the pallets cannot be retrieved at all. A more detailed description of the Optilogx box will be given in paragraph 2.3.

For the different warehouse layouts the amount of pallets that can be stored per square meter can differ allot, this can also be seen in table 2.1. If the amount of volume taken up by the warehouse is not an issue and time is not an issue as well, then all of the five layouts discussed above can be good options for the implementation of a warehouse. However, when minimizing the size of the warehouse and also minimizing the amount of time necessary to get the requested pallets out of the warehouse, then narrow aisle racking, push-back racking and a fully automated warehouses are the only options left. If the warehouse does not work according to the LIFO principle, only narrow aisle racking and fully automated warehouses can be considered. Figure 2.8 gives the comparison between the classical layout, the narrow aisle layout and the fully automated warehouse layout (Optilogx box). The big advantage of the fully automated warehouse is the fact that more pallets can be stored per surface area, with a total advantage of 124% more pallets per level.

The conditions that qualify the most for the implementation of a fully automated warehouse are the minimization of surface area and the labor/cooling costs of the warehouse. The cooling and labor costs refer to the warehouses used for perishable goods (cold stores). As said before, for warehouses operating at low temperatures it can be a reasonable and cost-efficient solution to fully automate.

Table 2.1: Comparison of Classical, Narrow aisle and Optilogx warehouses

	Classical	Narrow aisle	Optilogx
Total number of pallets per level	296	440	**663**
Advantage in comparison with classical layout	-	+48%	**+124%**
Technical outfeed			

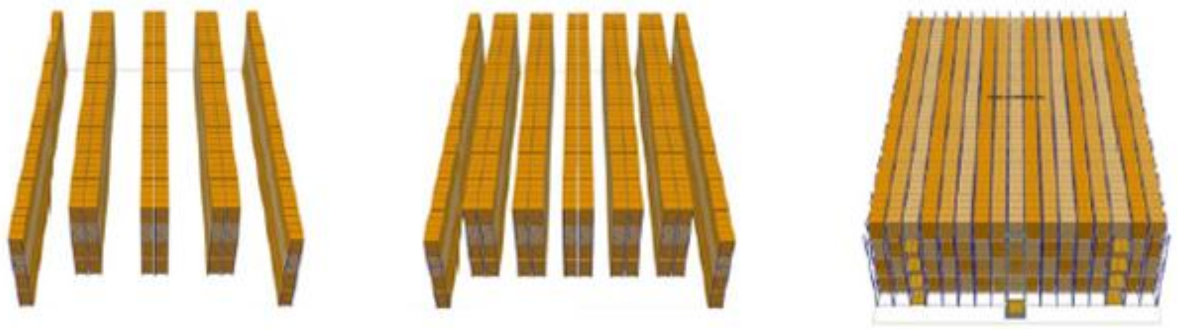


Figure 2.8: Classical, narrow aisle and Optilogx warehouses [1]

2.2 Other automated warehousing systems

2.2.1 Bicycle factory

The first comparable automated warehouse that is described is the automated warehouse of bicycle producer Gazelle[7]. This automated bicycle warehouse is located in Doesburg, about 4 kilometers from the Gazelle factory in Dieren. A truck drives about 20 times a day from this factory in Dieren to the automated warehouse. The main goal for this system was to reduce the damage of the bikes inside the warehouse. Especially paintwork damage was one of the main issues when storing the bicycles inside the old warehouse. The bicycles will be un-loaded at the infeed and moved into the automated warehouse. This infeed procedure includes hanging down the bicycles onto the front wheel of the bike (see figure 2.9). This hanging of the bicycles on the front wheel reduces the chance of damaging the bikes inside the warehouse.

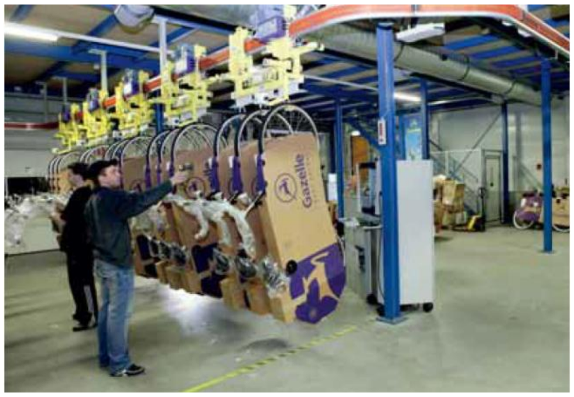


Figure 2.9: Damage reduction by storing the bicycles on the front wheel [7]

The storage inside the warehouse is operated according to a number of steps. When entering the warehouse, the bikes are hung up on the hooks of a trolley inside the warehouse and equipped with a unique and individual barcode. The trolleys are then moved to an area in which the contour check takes place. This contour check is done by electronic eyes that measure the bike and determine whether or not the bikes will fit into the warehouse. Then the bikes are transported to the actual storage component of the warehouse which is able to contain up to 30.000 bicycles. When arrived at the storage, robots will pick up the bicycles form the trolleys and divide them over 6 levels according to their badge information. When a truck requests a number of bikes this is known in advance. The requested bikes are moved out of the storage location to the outfeed of the warehouse, where they can be buffered till the correct truck arrives to pick them up. This buffering process mostly takes place at night, when there is no infeed and no personnel present inside the warehouse. The next morning all the requested bikes are ready at the outfeed to start filling up the trucks. The bikes needed in the first truck are also the first bikes in the prepared

[7] http://ttmnl.wpengine.netdna-cdn.com/wp-content/uploads/2007/01/07ttm12_disrep.pdf

buffer row and are ready to be loaded into the truck by hand. This preparation process provides the possibility to have around 4000 bicycle movements per day in- and out of the warehouse.

The control of the warehouse is done by a German company called PSB[8]. This company provides a LVS (a German warehouse management system) that is linked to a TMS (transport management system) from the company ROTRA[9]. The TMS 'knows' which batches come in, and when the batches get scanned at the infeed, also the WMS (LVS) inside the warehouse knows which batches have arrived. At that time the destination inside the warehouse is automatically assigned to the batch and the bicycles get moved to the desired location. With this same co-operation between the WMS and TMS, the outfeed is prepared and the right batches are provided when a truck arrives at the automated bicycle warehouse.

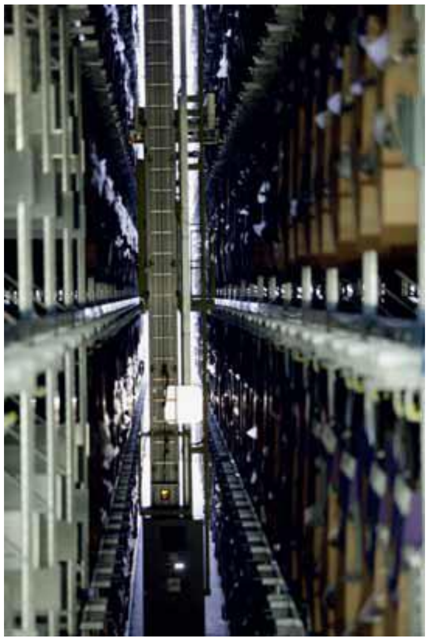


Figure 2.10: Aisle inside the automated bicycle warehouse [7]

Advantages/disadvantages

The big advantage of this system is the capability of preparing the outgoing orders for the next day at night. In this way the night-time at which no personnel is present inside the factory can be used really effective. Also the possibility to reach every bike inside the warehouse is an advantage of the automated system. However, this also means space is necessary to provide aisles for the trolleys to move between the racks, which leads to a bigger area usage of the warehouse. Another disadvantage is the loading and unloading of the trucks which still goes by hand. Therefore the speed at which the trucks are loaded, is still dependent of the personnel.

[8] http://www.psb-gmbh.de/

[9] http://www.rotra.eu/

2.2.2 Automated Car garages

Another fast growing market related to automated warehousing, is the market of the automated car garages. In more and more cities people can experience the possible future of car garages. However, this is not an entirely new concept. The first mechanical parking system was developed in Paris around 1906. The architecture and engineering office AG Perret established a kind of mechanical parking system with the garage on the Rue du Ponthieu[10]. This mechanical system worked by making use of a multi-story building with a vertical car elevator inside, to transport the cars to a higher level on which attendants parked the cars. Nowadays more and more fully automated parking garages arise in all different models and sizes. All these parking garages have the big advantage of decreasing the area needed for parking spaces. This necessary space can be limited, because no allowances need to be made for driving the car into the parking space or for opening the car doors. Also no driving lanes or ramps are needed for driving the car in and out of the parking garage. Another area reduction inside the automated parking garage is the height of the ceilings. No pedestrian traffic is present inside the garage, so the ceiling height can be minimized. The absence of passengers also eliminates the necessity to install walkways, lifts and stairways inside the parking garage. All these area reductions lead to the possibility of storing more than twice as many cars in the same amount of space as a regular parking garage. There are more advantages of these systems, for example the higher security inside the garages, as well as the parking damage which can be totally eliminated. While all these advantages contribute to a modern solution, also some problems arise when implementing a fully automated parking garage. What happens for example if 10 cars are requested at the same time? The normal waiting time for a car differs from 1 to a maximum of 3 minutes. This is still the case for every car inside the system, but customer number 10 still has to wait for all the previous cars. This can take a while, dependent on the type of system and the number of outfeed points inside this garage. Another big disadvantage to take into account is the fear of breakdown. What will happen if the system fails and the cars cannot be retrieved from the system? This is not just a fear of potential users of the system, but this already occurred multiple times in the last years. For example the case in Den Bosch where the automated car garage not only broke down, but also damaged a lot of cars[11]. The system just didn't work the way it was designed, mainly because of failing software. Therefore the municipality of Den Bosch decided to get rid of the automated car garage and rebuild the garage into a traditional one with drive ramps. This led to a reduction by half of the parking spaces, but (depending on the driver) people where able to retrieve their car without any damage.

From the years that the first (partly) mechanical automated parking garage was developed, lots of new concepts have arisen in the world of automated parking garages. Some of these fully automated systems have some resembles with this research project and these systems will therefore be discussed. Because of the wide variety of automated parking systems, not garages will be discussed. Only the automated garages that are closely related to this research project will

[10] http://automatedparkingsystems.blogspot.nl/2013/06/automated-car-parking-history.html
[11] http://www.computable.nl/artikel/nieuws/overheid/2255264/1277202/garage-plet-auto.html

be elaborated on. These garages are divided into two groups: the mono-path automated car garages, and the multi-path automated car garages.

Mono-path automated parking system

The first generation mono path automated parking systems[12] consist of rolling hoists that follow a rails mounted on the floor and the ceiling of the garage. This mechanical system provides the movements of the cars inside the garage. The individual movements made inside this system are relatively fast, but the system is not capable of handling any peak demands, so queues form instantly if more than one car is requested or offered to the system. This is a big disadvantage because this makes the system impossible to use in any environment that has peak and off-peak demands (for example offices). Another disadvantage is the amount of aisles needed inside a mono-path parking garage. This can be compared with a narrow aisle warehouse described in paragraph 2.1.

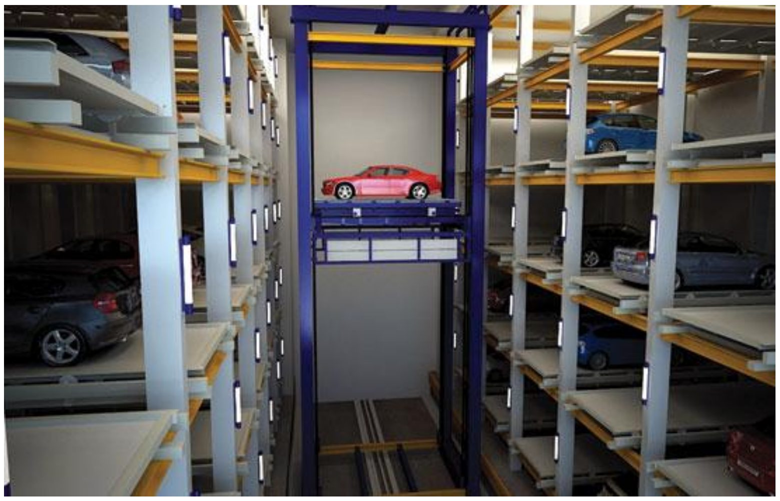


Figure 2.11: Mono-path automated parking [12]

A similar kind of system is the round automated parking system: the Autostadt CarTowers in Wolfsburg Germany[13], which can be seen in figure 2.12. This system works in the same way a mono path automated parking system works, only the area in which the cars are stored is of cylindrical shape. As can be seen in the figure there is one main rail constructed in the middle of the system. Along this rail robotic arms move up and down to put the cars in and out of the racks. Again a big disadvantage is the handling of peak demands in the system which will almost instantly lead to queues. This system is used for the storage of new Volkswagen cars. Customers can come in and retrieve their newly bought car from the automated system. In this case queuing is not a real problem, the system is more focused on the advertisement function and

[12] http://continuingeducation.construction.com/article_print.php?L=316&C=942
[13] http://www.thecoolist.com/autostadt-automated- parking-garage-towers-2/

the experience of the customer when picking up their new car. For these functions the system is really suitable.

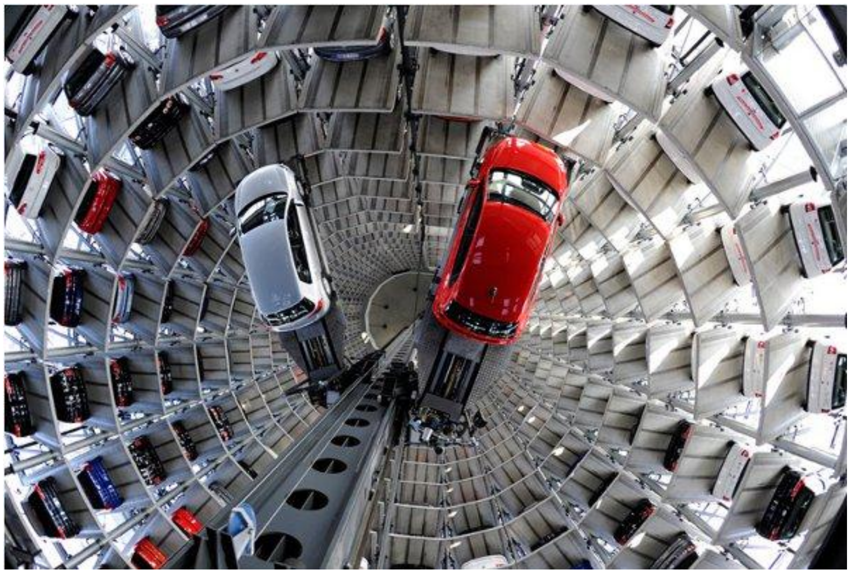


Figure 2.12: Round automated parking [13]

Multi-path automated parking systems

A later generation system is the multipath parking system. This system makes use of automated guided vehicles (AGV) which are not constrained to lanes inside the garage, but they are free to move in the garage along multiple paths. This multi-path functionality makes it possible for the AGVs to work around obstacles, and hereby avoiding the creation of bottlenecks in the main transit aisle. This means that the system is able to work with several AGVs inside the warehouse at the same time and therefore be less sensitive for the forming of queues. Another advantage of using multiple AGVs are the problems that arise when one of these AGVs fails. In this case not the entire garage will fail, but the work will just be taken over by another AGV. This means there is no single point of failure like there is in the case of mono path automated parking systems. A disadvantage is still hat there has to be space available for the movement of cars inside the garage from and to the locations where the cars will be parked.

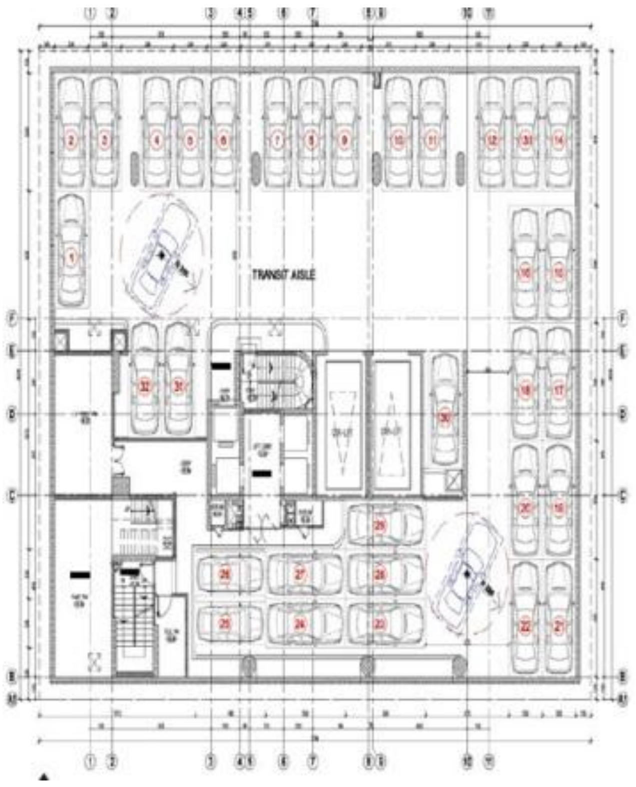


Figure 2.13: Multi-path automated parking [12]

These aisles for the movement of cars inside the garage are not necessary in the following Donhyang Optima Parking system[14] developed by Parkmatic. This system works with the loading of the garage from the top and the possibility to slide the lanes in which the cars are parked to the left and the right. Therefore every car can be reached without making use of aisles alongside every parking place. This system can be compared with the functionality of a sliding puzzle. Only one parking space has to be available to be able to reach any car inside the garage. This leads to big advantages in the respect of area usage. The big disadvantage however, is the time it can take to retrieve a car from this system.

[14] http://www.parkmatic.com/#!___automated/optima-parking

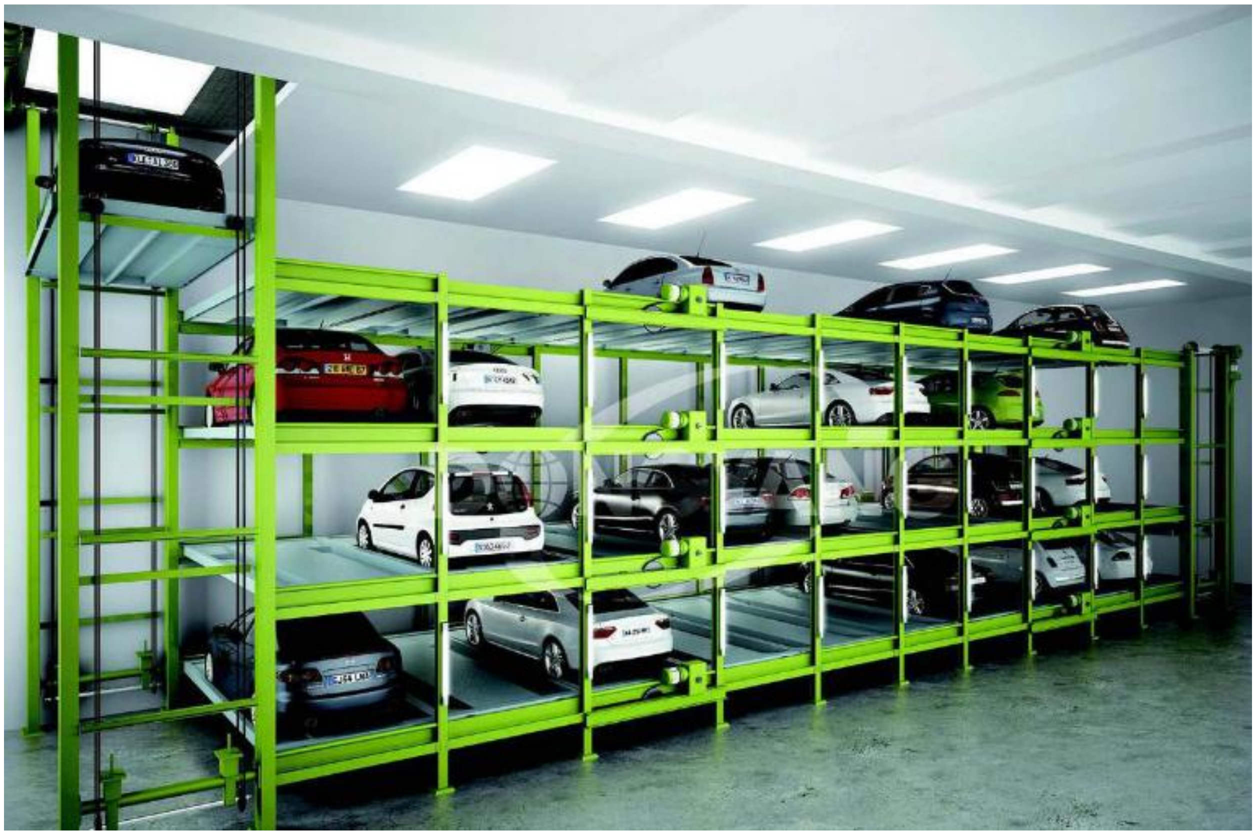


Figure 2.14: Automated car garage without aisles [14]

2.2.3 I-Cube

The last and most similar system to the already mentioned Optilogx box developed by ORTEC and SPIE, is the so called I-Cube. This system was realized in a co-operation of WICS solutions[15] (software) and Storax nv[16] (mechanical structure) and has the possibility of storing a large amount of pallets on a small surface area. It also has the possibility of high in- and outfeed speeds. This is the same combination of possibilities ORTEC and SPIE are offering to the customer. Figure 2.15 shows a configuration of the I-Cube, in which the storage racks and the mechanical components can be seen. The main resemblance with the Optilogx box is the use of chain-conveyers inside the buffer lanes of the warehouse. Therefore the lanes itself are able to move the pallets. One of the main differences with the Optilogx box is the control software of the I-Cube. The I-Cube makes use of a WCS (warehouse control system) that is able to control the different PLC's (Programmable Logic Controllers) inside the warehouse. A similarity with the software that ORTEC has developed to control the warehouses, is the simplicity of the software in the PLC's. This means the logistical complexity is for the biggest part calculated in the WMS software. This is almost the same situation as in ORTEC's software. The software inside the PLC's does not take

[15] http://www.wics.nl/

[16] http://www.storaxsupplies.com/

care of any of the complex calculations, this is all done by the overlaying software. In the software of ORTEC this is done with two layers on top of the layer of PLC's (more about working principle and the layout of the software in chapter 2.5)

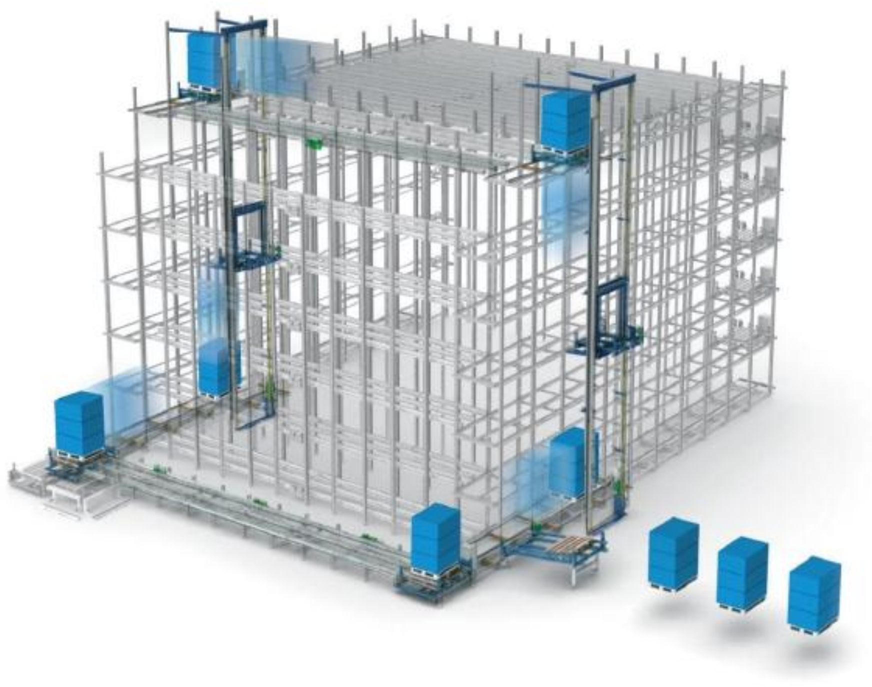


Figure 2.15: I-Cube [15]

2.2.4 Resemblances with the Optilogx box

In the paragraphs above some automated parking garage options are described. While these systems are not exactly the same as the automated warehousing projects ORTEC is working on right now, there are some similarities in these systems. Firstly the automated garages which work according to the mono-path principle are discussed. These garages need to be able to reach every single car on any time and therefore make use of aisles inside the garage, which has a negative effect on the amount of cars that can be stored inside the garage. This automated system is comparable with the automation of a narrow aisle conventional warehouse layout; the process is automated but the amount of extra storage space inside the warehouse is not optimally used. In the case of multi-path garages the effective use of storage space inside the garage can be used more optimally. The working principle of this system has a lot of resemblance with the Optilogx box that ORTEC is working on right now. Especially the option for automated car garages without the use of aisles is comparable with the Optilogx box, because the principle of the sliding puzzle is present in both the systems (see also the Optilogx description in the next paragraph). The system that has the most resemblance with the Optilogx box in terms of design, is the I-Cube discussed above. This system is designed to use chain-conveyers and shuttles to transport pallets inside the warehouse, and uses vertical transporters to move the pallet from one floor to another. The big difference with this system and the Optilogx box is the way this system is controlled (see also paragraph 2.5: how does the optimizer work).

2.3 Optilogx box description

In cooperation with SPIE Building-Systems, Nico van Rijn invented a system that focusses on optimization of the intake of (frozen) goods, the storage of pallets inside the warehouse, the physical handling of pallets and the sorting of pallets that have to be loaded into a trailer. The total solution involves the cooperation between SPIE and ORTEC. SPIE is responsible for delivering the mechanical/electronic parts for the Optilogx box, and ORTEC is responsible for the control software. One of the big advantages of this system is the high throughput rates it can handle and the sorting of the pallets inside the box, in order to get the pallets out in the right sequence. This reduces the handlings that are necessary by the pallet-truck drivers that load the truck. The loading of the trucks still goes by hand. The dimensions of the Optilogx box setup can be configured according to the need of the customer. The length, width and height of the box are entirely configurable. These changeable dimensions can lead to Optilogx box configurations with the exact pallet spaces available inside the box, equivalent to the customer's needs. The Optilogx box can be configured to work according to different inventory methods, for example: LIFO (Last in first out), FIFO (first in first out) or FEFO (first expired first out). The cold store warehouses located near a production plant, on which ORTEC is focused right now all make use of the FIFO principle.

In the next paragraphs the Optilogx box system will be described with the most important mechanical parts and the way these parts work together.

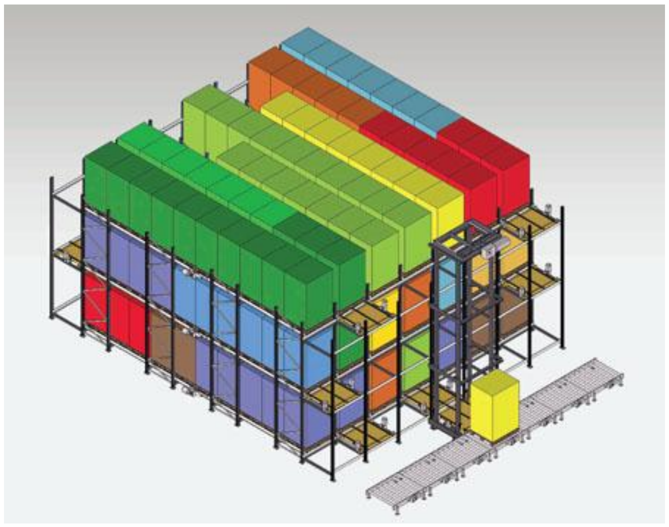


Figure 2.16: Optilogx box [1]

The Optilogx box consists of several mechanical parts from which an example of a configuration can be seen in figure 2.16. The big difference with conventional warehousing types, are the movable storage lanes and the transporter shuttles that make it possible for the warehouse to operate without the use of pallet-trucks inside the warehouse. This means no physical handling is necessary by the personnel to move a pallet from one location to another inside the warehouse. An example of one of the moveable storage lanes (also called buffer lanes) can be seen in figure 2.17. These storage lanes each consist of 3 chain conveyers on which the pallets are stored, and serve as the transportation for pallets inside a lane. The chain conveyers can be moved back and forth in order to transport the pallets to the requested direction. The movement of the lanes is made possible by an electric motor installed alongside the lanes. Every lane consisting of three chain conveyers is served by one electric motor. This can also be seen in figure 2.17. Dependent on the size of the Optilogx box, more than one of these lanes can be installed behind each other to create longer lanes. This is done at the production plant of Hellendoorn, in order to create buffer lanes with a length of 21000mm or 21 meters.

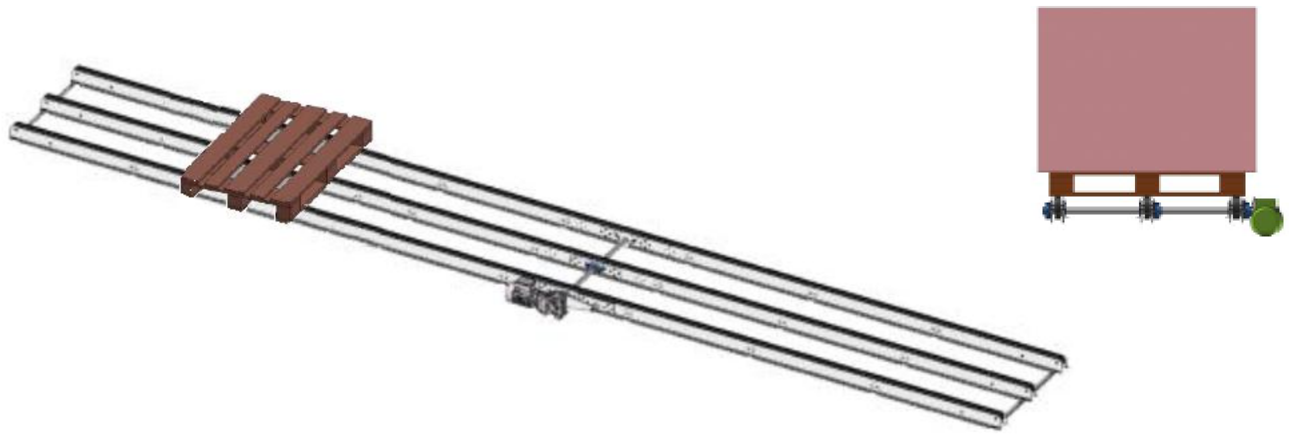


Figure 2.17: Buffer lane [1]

The Optilogx box is constructed with several of these buffer lanes connected with an aluminum construction. According to the demands of the customer the number of lanes inside the warehouse is determined. In order to be able to move the pallets from 1 buffer lane to another, two mechanical transporters are used in the box. The first transporter is used for the horizontal displacement of the pallet and is therefore called the Horizontal transporter (HT) or shuttle, see figure 2.18. These shuttles are placed on a certain level inside the box and can only move on that specific level. In order to move the pallet in the vertical plane, the box makes use of a vertical transporter (VT) (figure 2.19). This mechanical transporter can be compared with the working principle of a passenger lift. Instead of the persons that can walk in and out of the elevator themselves, the vertical transporter is equipped with 3 chain conveyers that make it possible for the pallets to be transported in- and out of the vertical transporter.

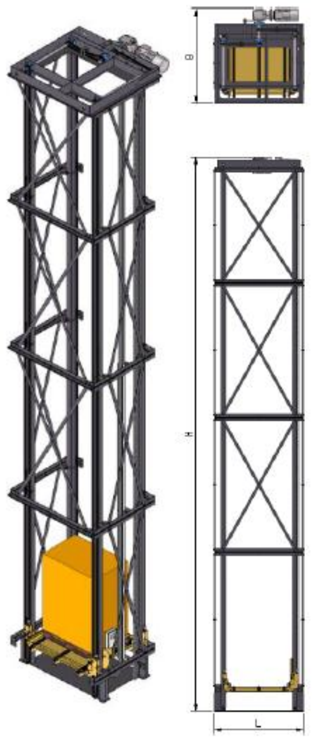


Figure 2.18: Vertical Transporter [1]

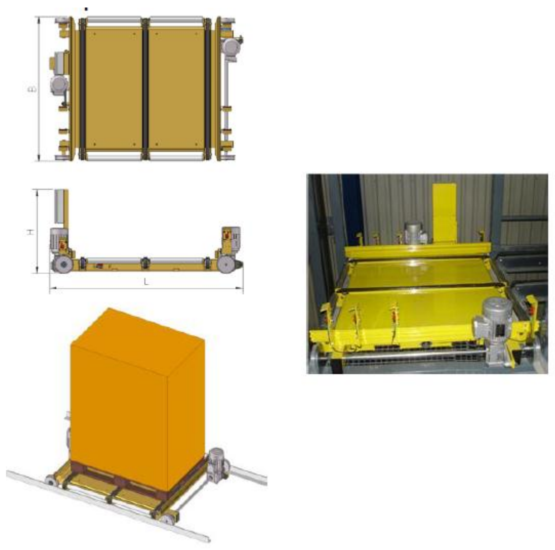


Figure 2.19: Horizontal Transporter or Shuttle [1]

Figure 2.20 shows how all the different parts described above are connected inside the Optilogx box. The pallets are shown in different colors inside the buffer lanes. Where the in- and the oufeed of the Optilogx box is positioned, is dependent on the demands of the customer. For example: the configuration of an Optilogx box in combination with a satellite warehouse, the infeed of the box will use the same vertical transporters as the outfeed of the warehouse. The same amount of vertical transporters will be positioned at the opposite direction of the in- and outfeed and will be used for the movement of pallets from and to the connected satellite warehouse. At the warehouse in Hellendoorn, the infeed is positioned at the back of the Optilogx box and the outfeed at the front of the Optilogx box. This is a logical position for the in- and outfeed because the production plant works according to the first in first out (FIFO) principle. This means that the product with the oldest production date will be selected to go out of the warehouse, if this kind of product is requested. This principle works the same for every other product type inside the warehouse and will prevent the expiration of the products.

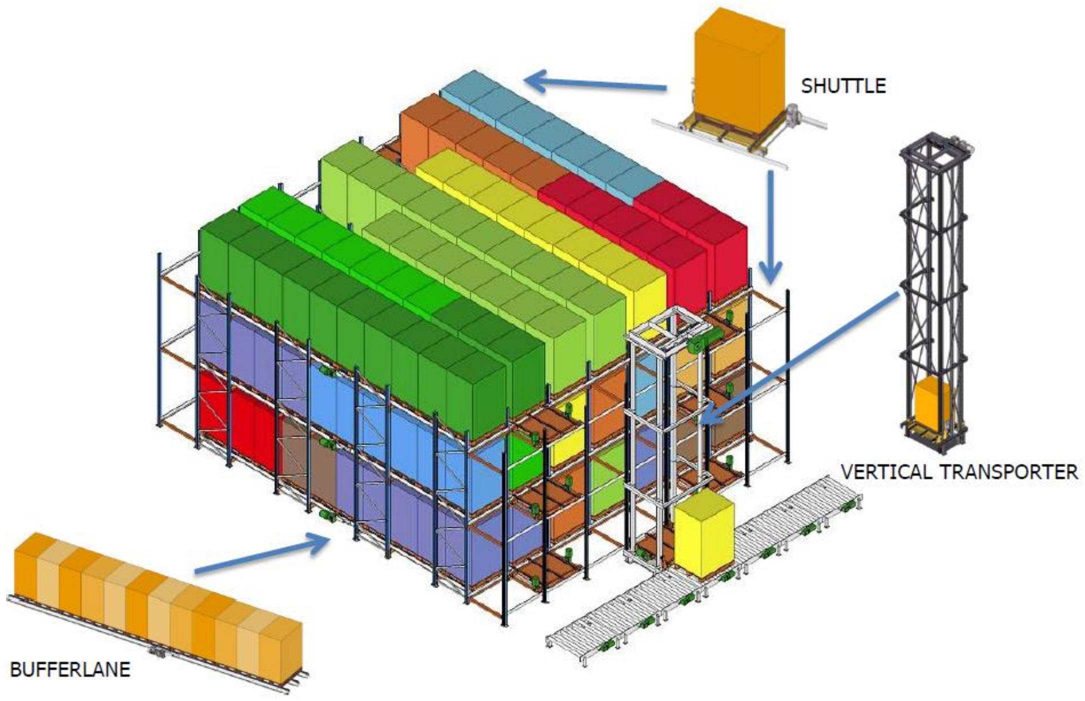


Figure 2.20: Optilogx box

2.3.1 Advantages of the Optilogx box

As can be seen in figure 2.20 of the previous paragraph, the pallets inside the buffer lanes of the Optilogx box, can be positioned really close together. In reality the gap between the different pallets in a buffer lane is around 50mm. This in order to prevent the pallets from bumping into each other, leading to damaged pallets or other problems.

The principle of this system has some resemblance with a sliding puzzle[17]. The difference is the number of degrees of freedom inside the box in comparison with the sliding puzzle. The big comparison is the situation in which only one pallet place is available inside the warehouse. In this situation it is still possible to retrieve any requested pallet. It may take some time however, if this requested pallet is positioned somewhere behind a lot of other pallets. The time it takes to retrieve the correct pallet can also be clarified with the example of a simple sliding puzzle: compare the situation of a sliding puzzle with only one empty place to reposition every piece of the puzzle, with a sliding puzzle with 5 empty places to move the pieces around. The second puzzle is much easier to solve in a far shorter amount of time.

[17] http://www.123rf.com

Figure 2.21: Sliding puzzle [17]

The width of the pallets that the buffer lanes can handle is 1200mm, this size is chosen to make it possible for the buffer lanes to handle both EURO and industrial pallets, also called CHEP pallets. As can be seen in the figure, the dimensional difference between EURO and industrial/CHEP pallets is the length of these pallets. The EURO pallet has a length of 800mm while the CHEP is 1000mm in length. The width of these pallets however is the same for both: 1200mm. This makes it possible for the buffer lanes to store both types of pallets combined in one lane. This leads to a wider variety of products that can be stored inside the warehouse, because some products are stored on CHEP and others on EURO pallets.

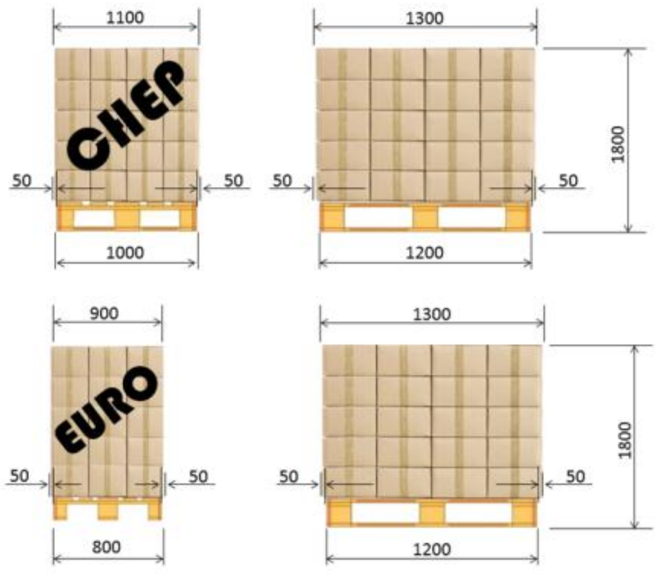


Figure 2.22: CHEP versus EURO Pallet

Another advantage of the Optilogx box is the possibility to keep track of products easily. Every product that arrives at the infeed of a warehouse gets scanned with a barcode scanner. From this scan all the information available for this pallet is communicated to the database of the warehouse. This database is used to sort and therefore store the pallets by product type. The principle of the Optilogx box is to keep these products sorted, and therefore make a relatively fast outfeed process possible. This sortation makes sure the pallet of a product that needs to go out first, is positioned at the front of a buffer lane and therefore the first one to be reached (the outfeed process will be further elaborated on in paragraph the next paragraph). The accessibility of the pallet information also makes it possible to keep track of the expiration date of the products inside the warehouse.

The entire process inside the Optilogx box is automated, which means there is no human interaction necessary to keep the warehouse running. For example if the production plant is running at night or in weekends, the warehouse can process the manufactured pallets from the infeed of the warehouse to the actual storage inside the warehouse. In the old warehouse this infeed procedure needed personnel to move the pallets from the infeed into the freezing cell, leading to the necessary presence of warehouse personnel anytime the plant is producing. Therefore no more personnel is needed at the warehouse in weekends, which limits the personnel costs when the warehouse only has to cope with the production of the plant (no outfeed will take place in weekends).

All these advantages are related to the main advantage of the Optilogx box: it makes a fast infeed and outfeed process possible, without making use of any human interaction to physically move pallets around in the warehouse.

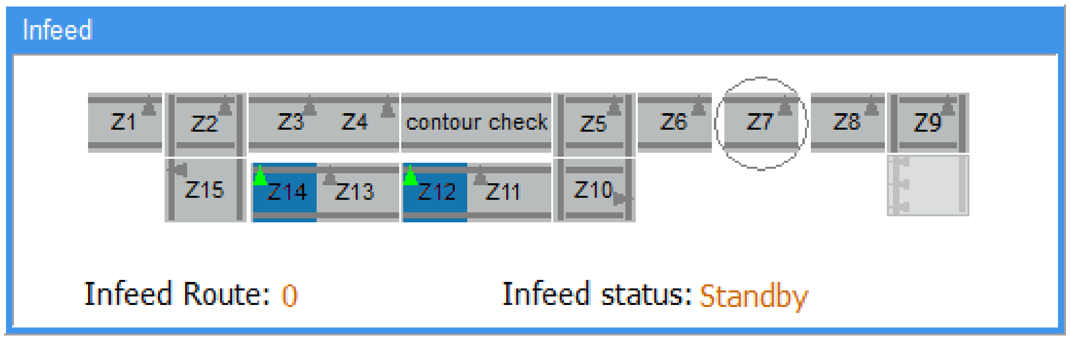
2.3.2 Connected modules

In the description of the Optilogx box in the previous paragraph, all the different mechanical parts are discussed. In order to make a specific design possible for every different customer, ORTEC and SPIE make use of a flexible design system. This design system consists of a customized setup of the box dimensions, but also of the possibility to connect different modules to the Optilogx box .These different modules can be connected and designed in the way the customer requires. Every module has its own function in the total automated warehouse and therefore the total automated warehouse can be adapted specifically to the functions it has to carry out. The modules described are not the only solutions possible, but just a sum-up of the current modules.

The Optilogx box can be seen as the heart of the system, from this starting point every other module is developed to fulfill the customer needs. The Optilogx box has this important function, because every pallet that comes in or goes out of the warehouse is transported through this Optilogx box. This means the box is involved in every in- or outfeed movement of any pallet. In order to divide the system into different modules, boundaries have to be determined for every separate part of the warehouse. The boundaries of the Optilogx box will be set on the points were pallets get in or out of the freezing cell. This determination of the boundary point is an important process, because in the rest of the report the focus will not be on all the different modules that can be used for an automated warehouse, but the focus will be on the module(s) that are most important in the entire process. The choice and explanation on which module(s) the focus will be on in this report will be given in paragraph 2.4: the technical capacity.

Infeed

Right before the pallets enter the Optilogx box, they are checked by a system that verifies whether the pallets have the right dimensions to go into the warehouse. This contour check can be seen as a separate system outside the Optilogx box. As set before, the boundary of the Optilogx system will be set right before the infeed and the outfeed. At the infeed, all the other parts necessary for supplying the Optilogx box with pallets can be seen as a separate system that works together with the Optilogx box. These parts consist of infeed lanes, which provide the infeed of pallets to the Optilogx box, the contour scanner, and reject lanes. These reject lanes are capable of storing the pallets that are rejected by the contour scanner. The figure represents the infeed lanes as configured at the production plant of the warehouse in Hellendoorn. The pallets go from the production on to the component Z1. Next the pallets are moved in the direction of the contour scanner in which they are checked. If approved they are moved in the direction of the Z9 which is connected to the warehouse, or they are rejected and will be moved down to the reject lane: Z10 till Z14. On every separate component there is a sensor installed with the function to register where the pallets are located. In this example, 2 pallets are present in the reject lane (on the Z12 and the Z14).



Infeed Route: 0

Infeed status: Standby

Figure 2.23: Infeed Hellendoorn

Outfeed

The same situation is present at the outfeed of the Optilogx box; there are different kind of systems available that take care of transporting the pallets from the outfeed of the Optilogx box to a location outside the cold store. Again, these systems are not part of the Optilogx box and therefore can be seen as separate components. At the existing warehouses with the OWCO controlling software, these outfeed procedures are either realized with outfeed lanes or with a mechanical component able to place the pallets on a predefined position.

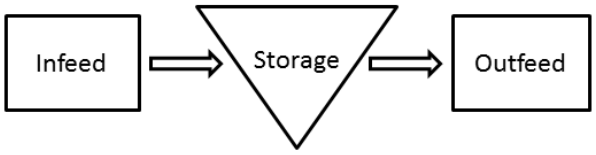


Figure 2.24: From Infeed through Storage to Outfeed

The outfeed lanes (comparable with the infeed lanes) are lanes from which a pallet truck driver can pick up the pallets. In the case of the warehouse in Hellendoorn the outfeed takes place by making use of the so called 'MUP' (in dutch: 'Multifunctioneel Uitlsag Punt'). This mechanical device places the pallets from the inside of the warehouse through a door, onto a pick-up place for a pallet-truck driver on ground level.

This sequence can be seen in figure 2.25, with the description of the different steps. The yellow rectangular device that moves the pallet from in- to outside the freeze cell, is the so called MUP.

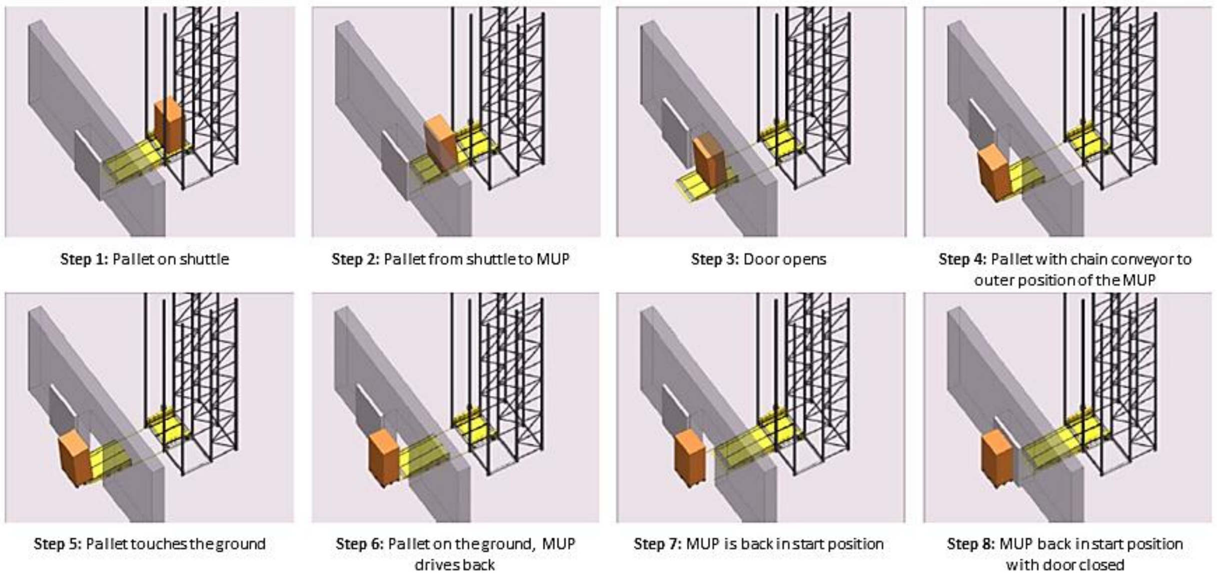

Step 1: Pallet on shuttle
Step 2: Pallet from shuttle to MUP
Step 3: Door opens
Step 4: Pallet with chain conveyor to outer position of the MUP
Step 5: Pallet touches the ground
Step 6: Pallet on the ground, MUP drives back
Step 7: MUP is back in start position
Step 8: MUP back in start position with door closed

Figure 2.25: MUP

Step 1: In this starting position the pallet is positioned on the Vertical Transporter (VT) which moved the pallet to the same level as the MUP. The VT is the yellow device inside the mechanical structure on which the pallet is positioned.

Step 2: The pallet gets moved from the VT onto the MUP by making use of the chain conveyers on both the MUP and the VT. These conveyers are driven at the same speed in order to guarantee a smooth transfer.

Step 3: In this step the door of the freezing cell opens and the MUP itself drives through the door on a rails. At the same time the conveyers on the MUP are still moving to transport the pallet down the angle (<5°) of the MUP.

Step 4: The pallet is moved by the chain conveyers to the outer position of the MUP.

Step 5: The chain conveyers will keep on turning until the sensors on the MUP will give the signal that the pallet is touching the ground.

Step 6: Once the pallet touches the ground, the MUP will drive back while keeping the chain conveyers turning in the exact opposite speed as the MUP is traveling in (see also figure 2.26). This will place the pallet on the ground with a small 10mm drop.

Step 7: Next the MUP will drive back inside the freezing cell to its starting position.

Step 8: The door closes and leaves the pallet outside of the cell, while the MUP is back in starting position ready to process the next pallet.

Figure 2.26 is a more detailed drawing of what happens in-between step 5 and step 6. In this process the speed of the MUP is marked by the blue arrow, and the speed of the chain conveyer is marked red. The important characteristic of this process is the equal but opposite speed of both the chain conveyer and the MUP. This prevents the pallet from damage because of sliding on the concrete floor.

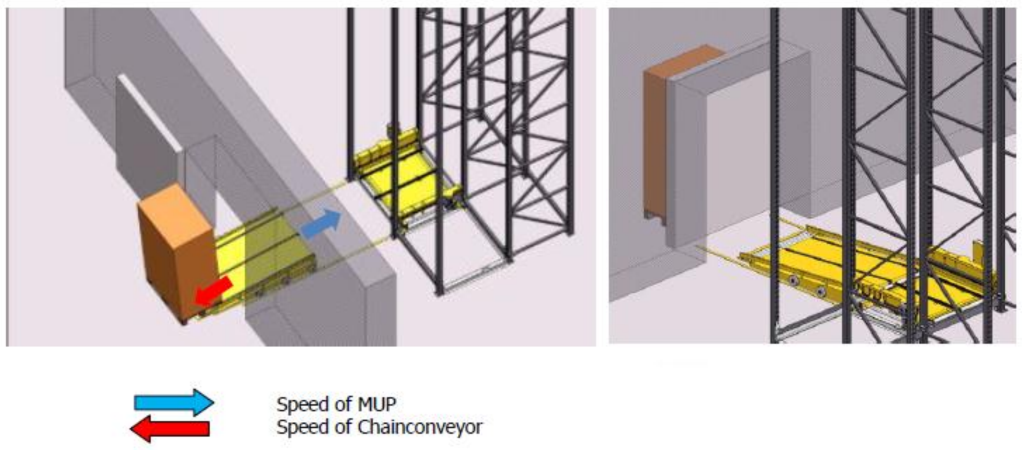


Figure 2.26: Working principle of the MUP

2.3.3 Storage

Another possibility for connecting the Optilogx box to other systems is to connect it to an extra storage unit. This storage is called the Optilogx-satellite warehouse or Optilogx-sat and has the function of providing secondary storage space as extension to the Optilogx box. This is the case at FreezeServe in the UK. The reason for this is to provide FreezeServe with a large storage space (nearly 13.000 extra pallet places are available in this Optilogx-sat), but still keep the advantages of the box (sorting and high outfeed capacity). In figure 2.27 an example of such an Optilogx-sat is given. The Optilogx-sat looks a bit like the Optilogx box itself with the main difference: the lack of chain conveyers inside the buffer lanes in this storage unit. This makes it possible to implement an Optilogx-sat warehouse for only a fourth of the cost per pallet place in comparison with the Optilogx box. The disadvantage is the fact that a lane itself cannot move the pallets back and forth anymore. Instead, the Optilogx-sat works by making use of small vehicles that are able to move the pallets in and out of the lanes. These small vehicles are called satellites and are able to lift up and transport the pallets in and out a lane. In order to move the satellite around in the Optilogx-sat, the system makes use of a sat-shuttle. This sat-shuttle can transport the satellite (with a pallet on it) to the requested lane inside the Optilogx-sat. The satellite will then drive the pallet into the lane and place it on the right position before returning to the Sat-shuttle. This same procedure can be carried out in reverse order to pick up a pallet from the Optilogx-sat. In the configuration at FreezeServe in the UK, one satellite and one shuttle are present per level in the Optilogx-sat.

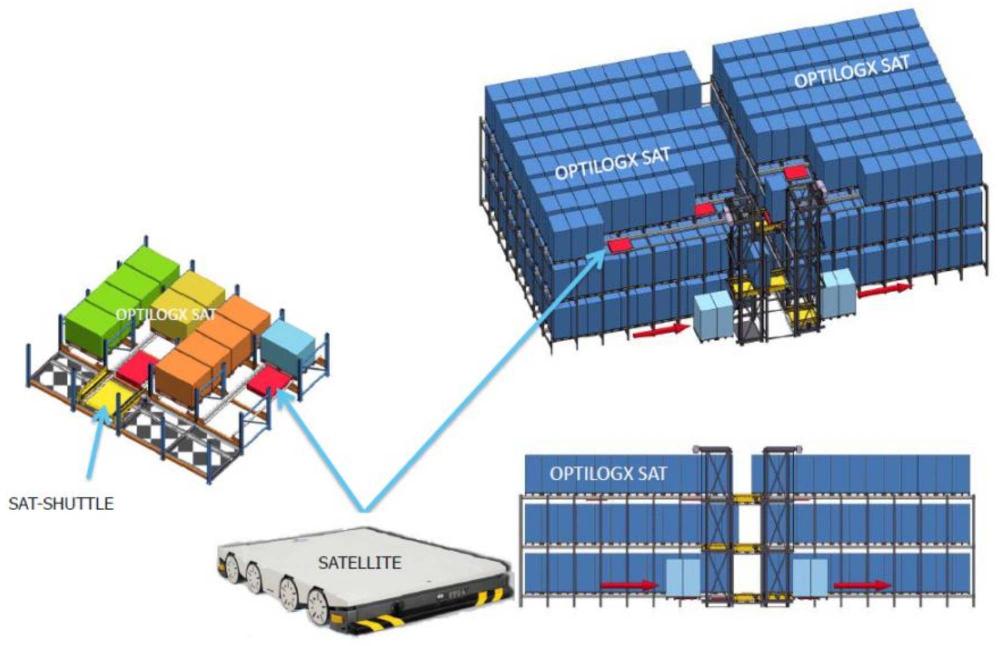


Figure 2.27: Optilogx-satellite warehouse [1]

2.4 Technical capacity

2.4.1 Capacity

According to the business dictionary[18] there are a lot of different definitions for capacity in a variety of contents. The definition of capacity related to manufacturing is: the highest sustainable output rate (maximum number of units per month, quarter, or year) that can be achieved with current resources, maintenance strategies, product specifications, etc. When looking at a warehouse, in most of the time the capacity refers to the total amount of pallets that can be stored inside. Therefore this (storage) capacity is dependent on the total storage space available inside the warehouse. This storage capacity can be determined in different kind of units, for example: the total amount of products storable inside a warehouse or the total amount of pallets that can be stored inside a warehouse. When combining the number of pallets that can be stored with the total area inside the warehouse, this capacity can be used as a performance indicator for the storage capacity. The units in which this storage capacity can be expressed, are the number of pallets per m^2, or the number of pallets per m^3. The different capacity requirements specified by the customer are dependent on the total (logistical) process the warehouse is part of.

Another performance indicator of a warehouse is the total capacity throughput. This throughput is determined by the amount of pallets that are able to be processed (in- or outfeed) by the warehouse. This maximum throughput can be seen as the technical (throughput) capacity of a warehouse and can be expresses by the equivalent of time (for example: number of pallets per hour). This technical capacity is one of the most important performance indicators of a warehouse, because this determines how many in- and outfeed movements are possible at a certain warehouse.

The maximum amount of infeed pallets that are able to be processed by the system is called the technical infeed capacity. This infeed capacity is dependent on the different resources at the infeed process of the warehouse. The technical outfeed capacity is also dependent on the resources at the outfeed process of the warehouse. In this specific case, the processes consist of the pallet movements in the warehouse that contribute to the outfeed procedure of the warehouse. In order to determine the technical outfeed capacity, the maximum number of pallets that can be moved out by the warehouse per time equivalent has to be determined. In order to do this, the bottleneck of the outfeed procedure has to be found. This bottleneck can be found by letting the system work optimally (as fast as mechanically possible). Hereby the assumption is made that the warehouse is totally sorted and therefore selected pallets will be moved from the front of the lanes to the outfeed. Subsequently the technical section limiting the outfeed process to work any faster has to be found. In other words, which mechanical part in the warehouse is keeping the other parts waiting? This section can then be determined as the bottleneck that limits the outfeed procedure. The maximum technical capacity then will be determined by the number of pallets moved to the outfeed when this 'bottleneck component' is working on maximum capacity.

[18] http://www.businessdictionary.com/

2.4.2 MUP

As described in paragraph 2.4, the MUP is a part of the warehouse at the Hellendoorn factory. This component takes care of the placement of pallets from inside the warehouse through a door to the outside of the freeze cell/cold store. The indication for the cycle time of the MUP is given in table 2.2, with the assumption that the pallet-truck driver removes the pallets on time. In the specific case of the production plant in Hellendoorn, the MUP can be seen as the bottleneck of the outfeed procedure and therefore determines the technical capacity of the warehouse.

For every full cycle, the MUP needs 43 seconds for every requested pallet. This leads to a total output capacity of maximum 80 pallets per hour. Off course in order to achieve this capacity, the pallet-truck driver needs to remove every pallet on time, but also the Optilogx box must be able to provide the MUP (with a pallet) every 43 seconds. If the Optilogx box is not able to provide the MUP with a pallet within these 43 seconds, the outfeed process will be delayed caused by the Optilogx box. These 43 seconds is determined by keeping track of the time the MUP takes for every pallet displacement when the system is running and taking the average. The deviation from these 43 seconds is relatively small (within one second), even the difference between moving a CHEP or a EURO pallet on the MUP differs less than a second.

Table 2.2: MUP cycle time

Action	**Time [sec]**
Pallet from OptilogX to MUP	10 sec
Door open	5 sec
MUP to outfeed position	10 sec
Pallet on floor	10 sec
MUP back to original position	8 sec (simultaneously the door is closed)
Total	**43 sec**

2.4.3 Time-lag

In order to achieve this technical outfeed capacity of 80 pallets an hour, the Optilogx box needs to provide the MUP with a pallet every 43 seconds or faster. Therefore the entire Optilogx box is involved in actually achieving the technical outfeed capacity. If anywhere in the Optilogx box the processes are not fast enough to stay within this maximum time of 43 seconds, delay arises on the outfeed procedure and the technical capacity will not be reached. The cause for this extra time or delay at the outfeed procedure can be defined as the time-lag inside the Optilogx box. In the ideal situation this time-lag or delay is 0. Therefore the MUP can process 83 pallets every hour at the outfeed of the Optilogx box.

Right now the reality is far from the ideal situation. Only in some specific cases this technical capacity can be realized. In all the other cases, the Optilogx box is not able to provide the MUP with a pallet every 43 seconds. In these cases the time-lag is tried to be kept as small as possible. The smaller the better because every extra second needed inside the Optilogx box to move the selected pallets to the outfeed, is extra waiting time for the pallet-truck driver. This directly result in extra waiting time for the 3PL's truck driver. There is not just one problem that keeps the box from performing inside the boundaries of the technical capacity, but there are many different situations that can lead to the logistical problem of time-lag and therefore outfeed delay inside the Optilogx box. Some of these situations will be discussed in the next paragraph.

In the case that another mechanical outfeed procedure is used instead of the MUP, this new outfeed procedure might have a much smaller processtime. An example can be given at the FreezeServe warehouse, where the outfeed is done by making use of chain conveyers which only have a processtime of several seconds. In this situation, another component will be the bottleneck inside the warehouse. Also an advantage is the possibility to store more than one pallet on a chain conveyer. From these chain conveyers the pallets can be picked up by the pallet truck drivers, who will limit the maximum outfeed capacity before the chain conveyers will.

If the pallet truck drivers can drop off a pallet faster than the Optilogx box can provide a pallet at the outfeed, the bottleneck of the outfeed procedure will have to be determined somewhere inside the Optilogx box. As already mentioned, this bottleneck will be the component that is limiting the other mechanical components to work at full capacity. In the case of FreezeServe these limiting components within the box can be defined as the vertical transporters. This is often the case if the outfeed is not the limiting factor inside the Optilogx box. The reason for this is the relatively low ratio of vertical transporters in comparison to the buffer lanes. FreezeServe makes use of an Optilogx-Sat, which means the products are not (all) stored inside the Optilogx box. In order to bring an entire order to the outfeed the selected pallets get prepared inside the Optilogx box. Therefore the box is used differently than at the Hellendoorn production plant. In the case of FreezeServe the shuttles which bring the selected pallets to the box are not able to keep the VT's busy the entire time. Therefore at the preparing of the orders, these shuttles can be determined as the bottleneck inside the entire system. This is not a major concern however: at FreezeServe the box is used in order to buffer the selected pallets and put them in the right order. Therefore the time it takes to prepare these pallets is has no influence on the outfeed procedure. They just need to make sure that the selection and preparation of the pallets starts on time.

2.5 How does the ‘optimizer’ work?

2.5.1 Layer by layer

The software developed by ORTEC to control the automated warehouses is called OWCO (ORTEC Warehouse Control and Optimization) and is build up in separate layers. These layers are able to communicate with each other by making use of text files (XML files) and databases. With this information the software is able to determine which pallets are inside the warehouse and on which places these pallets are present. In figure: 2.28 a simplified overview of the software structure is given. Using this overview, a description is provided for every layer of the software and the co-operation between these different layers. The communication represented by the blue arrows is further elaborated on in paragraph 2.6.

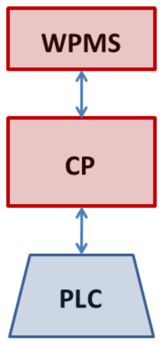


Figure 2.28: The different components of the OWCO software

Figure 2.28 gives 3 components, which can be divided into the three software layers. The red components at the top two layers, represent the main software programs developed by ORTEC to control the warehouse. The blue component on the bottom layer is an element of the warehouse that is controlled by the software of ORTEC. The layers in which the components are presented is similar to the layers in the actual software. The top layer provides the ‘brains’ of the software, the middle layer is for deciding which components need to be controlled and the bottom layer functions as the direct control on all the components.

The center of the figure represents the main software component, the program called CP. CP (Connactive Processes) is a collective name for several other smaller programs. As can be seen in the figure, CP is the center of the information flowchart. This means in every action that is done inside the warehouse, CP is involved. More detailed information about CP will be given in the next part of this paragraph. The most important connections to CP are displayed: on the bottom the connection to the PLC. The PLC is shown in the blue square and stands for Programmable Logic Controller. These are the connections from ORTEC’s software to the actual motors of the mechanical parts in the warehouse. If CP sends a command to the PLC, the PLC will transmit this command to the correct motors inside the warehouse.

Inside CP the database of the automated warehouse is located (see also figure: 2.29). This database contains all the information about the products that are currently in the warehouse, but also the products that have ever been inside the warehouse. The database also contains the lay-out information of all the current pallets inside the warehouse.

CP is then able to send this layout to the layer on the top of the figure: WPMS. WPMS stands for: Warehouse Planning & Management System. This program receives the layout of the current products inside the warehouse from CP, as well as the requests for pallets from the users of the system. This request is called an orderbatch, and normally contains between the 26 and 33 pallets. With this information WPMS selects the right pallets inside the warehouse and generates actions for the movements of these pallets. These actions will consist of the start and end position of a pallet involved in an action. WPMS can be seen as the top layer inside the software that functions as the 'thinking' component. These actions will be sent back to CP, which can be seen as the controlling component of the software. CP will pick-up these actions and will generate the routes belonging to these pallet displacements. In this case of this Optilogx box there is only one route from every lane to the outfeed. Therefore the generated routes will be similar every time for each specific origin-destination pair. Finally the connection from CP to the PLC (figure 2.28) makes sure the right motors inside the warehouse move, and thus the right pallets travel to the right location.

As stated above, the main component of the software (CP) is a collective name for several other smaller programs. Figure 2.29 gives a more detailed view of the components of which the total OWCO system is built out of. The big difference with figure 2.28 is that CP is now split up into separate programs. The way these programs are connected to each other and how CP is connected to the rest of the system, can also be seen in this figure.

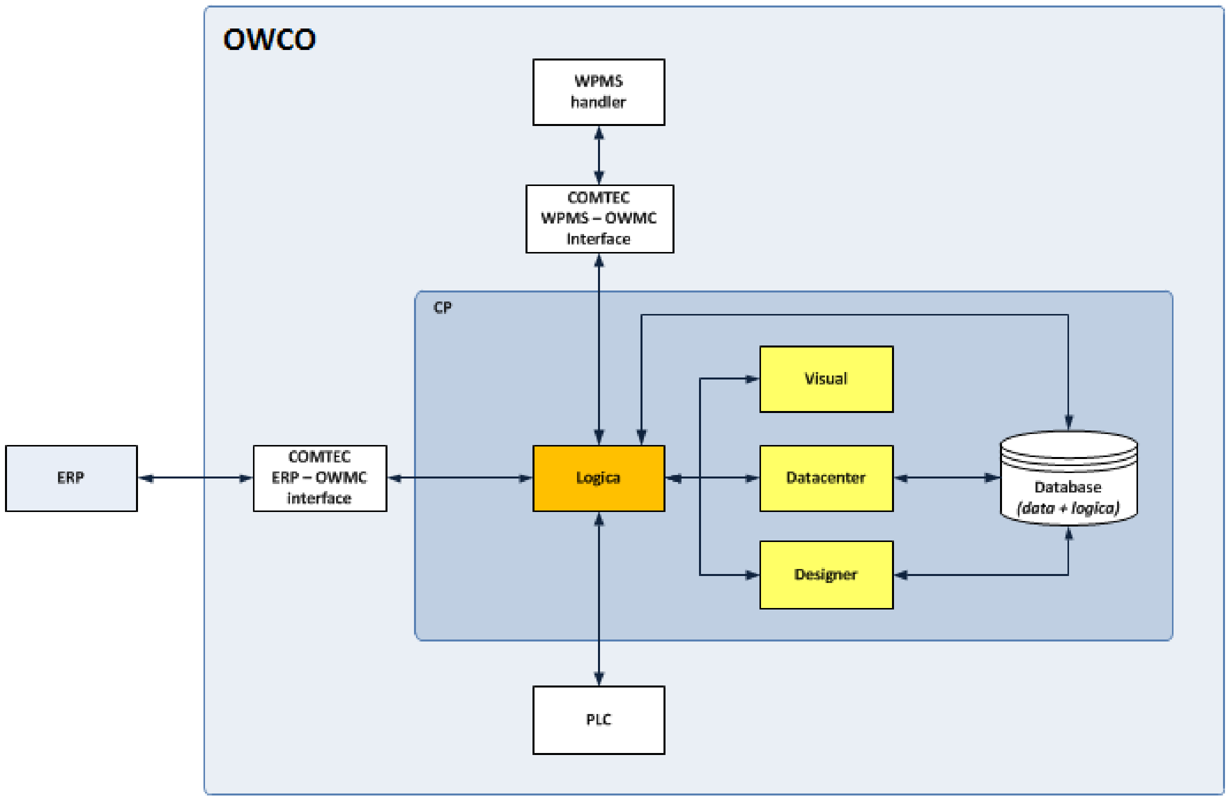


Figure 2.29: The OWCO software components

The big light blue square in the figure represents the boundary of ORTEC's software. Everything inside this big blue square is either controlled by ORTEC's software, or is a component of the software itself. The PLC's are the components controlled by the software of ORTEC, the rest of the components are interfaces or parts of the software.

The only part outside this boundary is the Enterprise Resource Planning system (ERP) that is used in the production plant of the customer. This ERP system is connected with ORTEC's software by making use of COMTEC. This is an environment that can be used for this kind of connections between an ERP system and controlling software. ORTEC uses this building platform in a lot of their software products. The function of the connection to the ERP system, is the possibility to provide the software with information of (upcoming) pallets, truckloads or which products are assigned to which batches, but also the other way around: to provide the ERP system with the up to date information of the warehouse (which batches arrived in or left the warehouse).

The ERP system is connected to the program called CP. This program is displayed in the figure by the smaller (bit darker) blue square inside the bigger one. As said before, CP is a collective name for several other smaller programs. These programs can also be seen in the scheme: Logica, Visual, Datacenter and Designer. Every program has its own functions, Logica is the engine of the system and does the actual calculations. The other programs can be seen as the tool for the user to control the warehouse. They have the same function as their names do suspect: Visual is the visualization of the warehouse for the user, Datacenter makes it easy for the user to retrieve data from the warehouse, and designer is the tool to make any changes in the software configuration of the system.

The last element connected inside CP is the Database of the warehouse (the white cylinder). In this database the entire inventory with pallet locations and history is available.

2.6 Communication between the different programs

As can be seen in figure 2.29 the PLC's are controlled by the CP program. This communication is done by an internet connection on the network available in the warehouses. The same way of communication takes place between the ERP of the production plant and the software of ORTEC: they are all connected on the available network in the warehouse. With this connection information can be send back and forth to CP, either coming from or going to the ERP system or the PLC's.

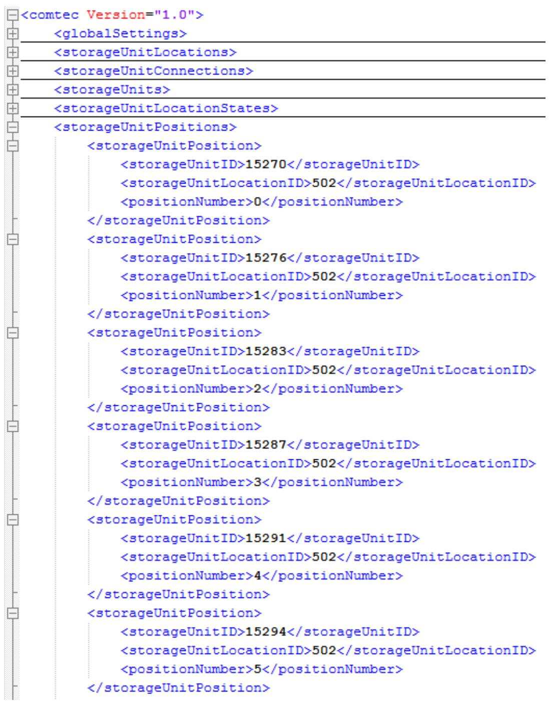
The communication inside CP is done a bit differently; the different parts of CP are all installed on the same computer. This makes it possible for the programs to communicate directly with each other and the database. From this database, Logica then collects all the necessary information and communicates this information to WPMS. This communication between CP and WPMS needs a bit more explanation than the previous more standard communication types. The communication between CP and WPMS is done by sending small text files back and forth between the two programs. These text files are called XML files (or Extensible Markup Language files), and can be seen as a set of rules for the encoding of documents that is both human-readable, but also usable by a computer. These XML files mainly consist of information about the positions of the pallets inside the warehouse and orderbatches that are requested by the users (for outfeed).

2.6.1 CPtoWPMS & WPMStoCP

CP has access to all the information inside the database. From this database CP sends an XML file with the current information from the warehouse to a certain folder on the computer CP is running on. WPMS is running on a server and is able to constantly access this folder. If an XML file is send from CP to this folder, WPMS immediately 'sees' this file and picks it up. WPMS then processes the file and with this information received from CP, WPMS can calculate which commands are necessary to fulfill the customers' requests. For the overview of the communication flow see figure: 2.32.

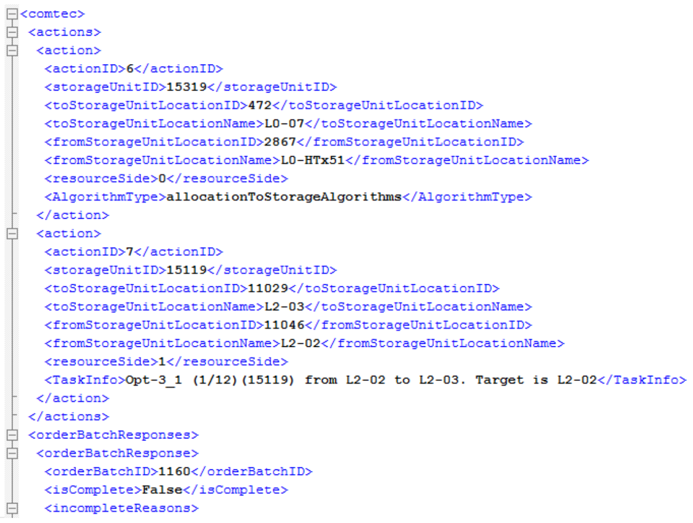
WPMS then runs this information through an algorithm and sends out an XML file with movements and other information as the result. This XML file will be send back to the computer CP is running on, from which they will be read by CP. From this information CP is able to determine which actions it has to send to the PLC's. These actions will then be sent to the PLC's which make sure the right mechanical devices (motors etc.) move the pallets to the right place. If a movement is done, this is again communicated through CP to WPMS. WPMS then calculates new movements and sends these movements to CP. More information about this process can be found later on in this paragraph.

The communication between CP and WPMS can be seen as a discrete signal, every communication step that has taken place can be counted as an XML file. The time window in which these communication steps can take place is continuous. In real time this means the XML files can be sent back and forth multiple times per second. WPMS and CP are constantly monitoring their target folders to 'see' whether there is a new file to process. If for some reason the files come into the folders faster than they can be read either by CP or WPMS, the files will just be stored in these folders. WPMS and CP will then always take the oldest file available in these folders. This to guarantee the right sequence of XML files is used; the last XML file sent out by CP is also the last file that needs to be read by WPMS. An Example of a file from CP to WPMS and from WPMS to CP is given in the figures 2.30 and 2.31:



```
<comtec Version="1.0">
    <globalSettings>
    <storageUnitLocations>
    <storageUnitConnections>
    <storageUnits>
    <storageUnitLocationStates>
    <storageUnitPositions>
        <storageUnitPosition>
            <storageUnitID>15270</storageUnitID>
            <storageUnitLocationID>502</storageUnitLocationID>
            <positionNumber>0</positionNumber>
        </storageUnitPosition>
        <storageUnitPosition>
            <storageUnitID>15276</storageUnitID>
            <storageUnitLocationID>502</storageUnitLocationID>
            <positionNumber>1</positionNumber>
        </storageUnitPosition>
        <storageUnitPosition>
            <storageUnitID>15283</storageUnitID>
            <storageUnitLocationID>502</storageUnitLocationID>
            <positionNumber>2</positionNumber>
        </storageUnitPosition>
        <storageUnitPosition>
            <storageUnitID>15287</storageUnitID>
            <storageUnitLocationID>502</storageUnitLocationID>
            <positionNumber>3</positionNumber>
        </storageUnitPosition>
        <storageUnitPosition>
            <storageUnitID>15291</storageUnitID>
            <storageUnitLocationID>502</storageUnitLocationID>
            <positionNumber>4</positionNumber>
        </storageUnitPosition>
        <storageUnitPosition>
            <storageUnitID>15294</storageUnitID>
            <storageUnitLocationID>502</storageUnitLocationID>
            <positionNumber>5</positionNumber>
        </storageUnitPosition>
```

Figure 2.30: CPtoWPMS communication file



```
<comtec>
 <actions>
  <action>
   <actionID>6</actionID>
   <storageUnitID>15319</storageUnitID>
   <toStorageUnitLocationID>472</toStorageUnitLocationID>
   <toStorageUnitLocationName>L0-07</toStorageUnitLocationName>
   <fromStorageUnitLocationID>2867</fromStorageUnitLocationID>
   <fromStorageUnitLocationName>L0-HTx51</fromStorageUnitLocationName>
   <resourceSide>0</resourceSide>
   <AlgorithmType>allocationToStorageAlgorithms</AlgorithmType>
  </action>
  <action>
   <actionID>7</actionID>
   <storageUnitID>15119</storageUnitID>
   <toStorageUnitLocationID>11029</toStorageUnitLocationID>
   <toStorageUnitLocationName>L2-03</toStorageUnitLocationName>
   <fromStorageUnitLocationID>11046</fromStorageUnitLocationID>
   <fromStorageUnitLocationName>L2-02</fromStorageUnitLocationName>
   <resourceSide>1</resourceSide>
   <TaskInfo>Opt-3_1 (1/12)(15119) from L2-02 to L2-03. Target is L2-02</TaskInfo>
  </action>
 </actions>
 <orderBatchResponses>
  <orderBatchResponse>
   <orderBatchID>1160</orderBatchID>
   <isComplete>False</isComplete>
   <incompleteReasons>
```

Figure 2.31: WPMStoCP communication file

In the total CPtoWPMS file (figure 2.30), **all** the information is available for the entire layout of the warehouse. The figure only shows a small part of the total information within this XML file. Every pallet that is physically inside the warehouse is communicated by this XML file. To give an example, the storage unit positions of some pallets are given (figure 2.30). Inside this storage unit positions, 3 subgroups are made per pallet. The first group is the Storage unit ID; this is the pallet number which corresponds to a certain pallet in the database. Then the storage unit location represents the lane in which the pallet is located, and the position number is the actual position inside the buffer lane. These 3 groups represent one storage unit position, and are given for every pallet inside the warehouse. Like this small example of only the storage unit positions of 6 pallets, the entire layout of the warehouse is present in this single XML file, consisting of the storage unit locations, the storage unit connections, the storage units and the storage unit location states.

This XML file with the layout of the warehouse only has to be sent a single time to WPMS. Once WPMS recieved the entire layout of the warehouse (including the pallets inside the warehouse), it only has to be updated on the changes that are made in the warehouse. WPMS is then able to determine the new warehouse layout itself. This updating of pallet positions inside the warehouse is realized with these same communication cycles using XML files.

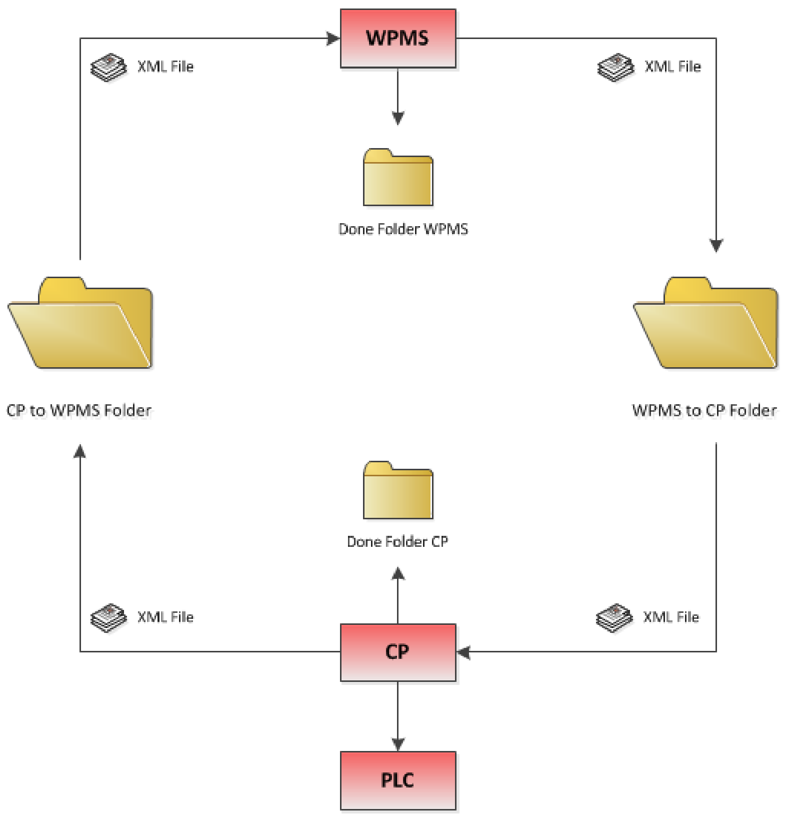


Figure 2.32: Communication flow

Once a file is read by either CP or WPMS, it will be removed from the target folder and be stored in another. This other folder is called the ‘done’ folder and contains all the XML files that are processed by CP or WPMS. Every action generated by WPMS and every action done by CP can be seen in these ‘done’ folders, also all the actions that are done in the past.

Figure 2.32 is a representation of the information flow from CP to WPMS and back. The flow starts at CP in the lower middle of the figure. The first step for CP is to make an XML file from the database with the entire layout of the warehouse. This XML file is then sent to the CPtoWPMS folder. WPMS constantly monitores this folder to see there are any XML files inside this folder to process. This folder can be seen as the infeed of the WPMS program. WPMS then processes the XML file that was sent by CP and generates a new XML file containing actions for CP to perform. It also sends a copy to the ‘done folder’, which contains the history of all the XML files. The generated XML file is sent to the WPMStoCP folder, this is the folder that is constantly monitored by CP. Once the XML file from WPMS arrives at this folder, CP immediately picks it up and processes it. This processing finally leads to actions for the PLC’s inside the warehouse. The PLC’s returns feedback to CP which makes the whole cycle start again. As said before, this flow is a continuous process.

2.7 Working principle of the Optilogx box

As described in paragraph 2.4, the technical capacity is the maximum outfeed capacity of a warehouse dependent on the bottleneck of the entire system. In the case of the Hellendoorn plant the bottleneck of the technical capacity is determined by the MUP. This MUP has an outfeed speed of 1 pallet every 43 seconds, which leads to a technical capacity of 80 pallets an hour. In order to reach this technical capacity the MUP has to be supplied with a pallet within these 43 seconds. If a pallet is supplied later than 43 seconds, time-lag is present and the 80 pallets an hour can no longer be achieved. There are many reasons for time-lag to originate in the system. The following paragraph will give some common examples of situations in which time-lag in the system starts to arise, as well as the reasons for this time-lag. In this case the examples will be given by making use of the Optilogx box configuration as present at Hellendoorn. However, these cases can also occur at a different configuration of an automated warehouse with the Optilogx box.

2.7.1 Main principle of the Optilogx box

In order to explain where the time-lag inside the Optilogx box originates, first the working principle of the box should be explained. What does the ideal starting position of the box look like, and how should this starting position be realized. First of all, what is the ideal starting position of the box? This should be a position in which the pallets are located in the box in such way that the pallets needed first, are also the ones at the front of the buffer lanes closest to shuttle that brings the pallets to the exit of the warehouse. When looking at the different products inside the box, the first pallet entering the warehouse of a certain product is (normally) also the first one that comes out. A small conclusion can be drawn about the working principle of the box: it works with a first in first out (FIFO) principle on product level. This can also mean that the oldest pallet of one product is way younger than the oldest product of any other. In the ideal situation, the box is sorted exactly according to the FIFO principle with the different products on separate lanes. In this case the box should be able to select the needed pallets for a product at the front of the lanes and immediately transport these pallets to the outfeed.

2.7.2 Different situations

In the ideal situation described in the previous paragraph, there should arise minimal to no time-lag inside the box. However, this ideal situation is not the most common situation in the Optilogx box. In most cases when an orderbatch is requested, there are some pallets needed that are not at the first position inside a buffer lane, because there are situation where there are just not enough lanes to provide a good position for every pallet inside the warehouse. For example if there are more different products inside the warehouse as there are lanes. Another point of focus is the fact that not every pallet of a single product has the same amount of product colli (in this case the number of ice cups) on every pallet. Therefore when an orderbatch is selected which requests an exact amount of colli, sometimes one or more pallets have to be dug out from behind a row of other pallets. This process can be seen in the figure 2.33 and 2.34 which represent the lay-out in the Visual of the production plant in Hellendoorn. The ground floor is presented (level 0) with the outfeed MUP on the bottom of the figures.

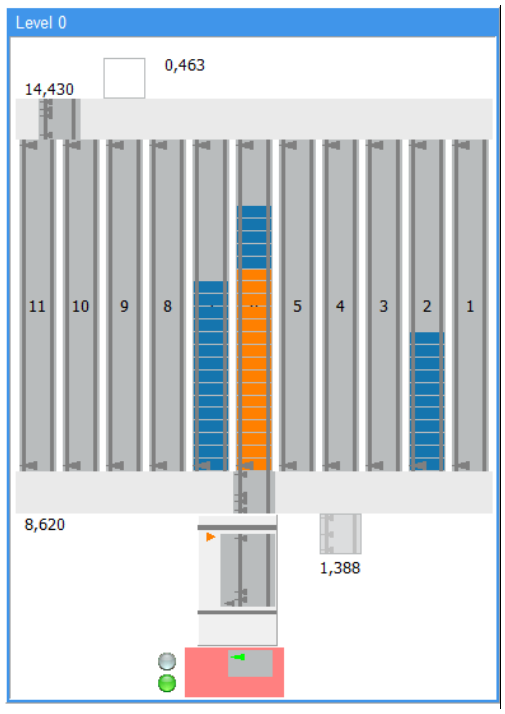


Figure 2.33: Ideal situation

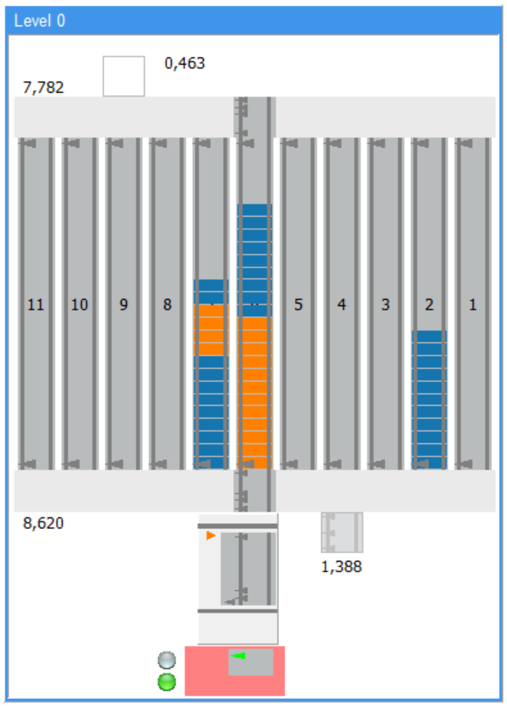


Figure 2.34: Not ideal situation

On figure (2.33) the ideal situation can be seen, where the blue small bars represent pallets and the orange small bars are the selected pallets. The grey elongated bars are the chain conveyer lanes (with the selected orange pallets on lane 6), the grey squares represent the horizontal and vertical transporters, and finally the big grey square on the bottom represents the outfeed MUP and the droppoint in red. As can be seen this is level 0 (the ground level) of the Optilogx box. The total box consists of 4 levels, all built up of 11 lanes. In the ideal situation the selected products are at the front of the lane, and can easily be picked out of the lanes and be transported to the MUP. No time-lag will arise in this situation.

In figure 2.34 the situation is presented where not all the selected pallets are at the front of a lane. Most of the pallets are again in front of lane 6, but there are also some pallets selected on lane 7. In this case the selected pallets in lane 7 are not the first ones in the lane, and therefore the system will have to dig-out these pallets located behind the non-selected pallets. The pallets will be moved to the nearest lane with empty pallet positions available. The box will need some time for this, and therefore the question arises whether or not the Optilogx box will be able to provide the MUP with a pallet every 43 seconds.

The problem when this situation arises is that WPMS, responsible for generating the actions, does not take the time into account in any calculation. WPMS does not have any 'knowledge' of resources or (process) times. This leads to the generation of a job-sequence which is far from optimal. An example of such a situation is given in figure 2.35 till 2.38, where pallets are moved to the outfeed from level 0 and level 1. In this example the sequence of the jobs generates an amount of time-lag that is directly affecting the outfeed procedure.

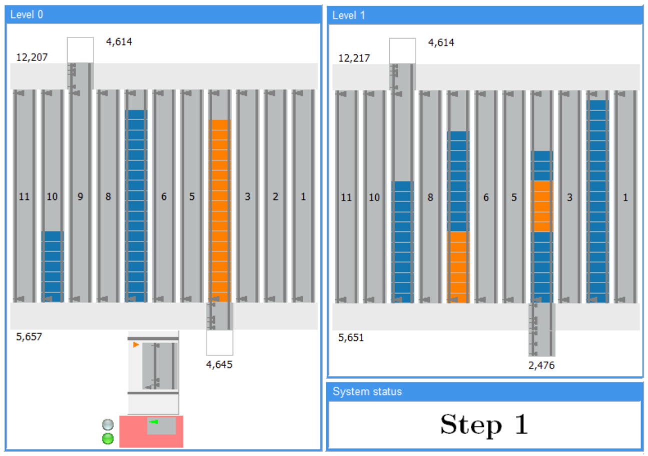


Figure 2.35: Step1: Start situation on Level 0 and 1

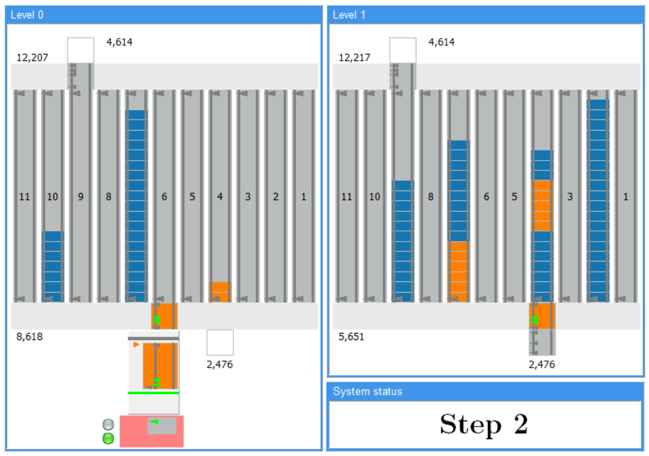


Figure 2.36: Step 2: Pallets to outfeed on Level 0

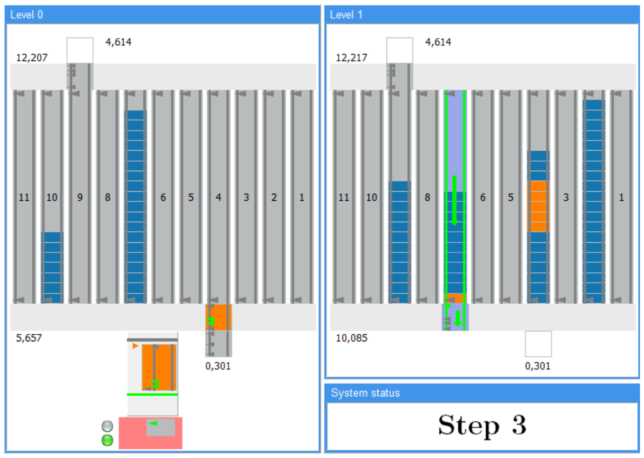


Figure 2.37: Step 3: Pallets to outfeed from Level 1

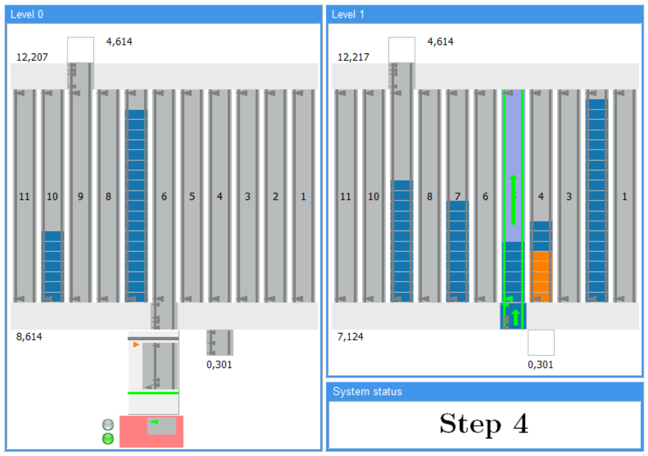


Figure 2.38: Step 4: Last pallets to outfeed from Level 1

In the figures it is clear that the optimizer first choses to transport all the pallets on L0 to the outfeed. In the meantime actions are created for the pallets on level 1 lane 7 to be moved down to the outfeed. This leads to the situation in which the pallet on the shuttle on level 1 is waiting to get moved down to the outfeed and is keeping the shuttle on level 1 busy. Because this shuttle is busy, the pallets that need to be dug-out (lane 4) have to wait for the shuttle to be available.

The first pallets from level 0 will be moved to the outfeed without any problem (step 2), no time-lag arises. Then the pallets from level 1 will be moved to the outfeed, again without any time-lag (step 3). But when the pallets of level 1-lane 4 need to be dug out, the MUP has to wait for this process and therefore also the truck-driver has to wait. This process caused waiting time in the system and therefore time-lag arose. In step 4 the pallets are dug-out and they will be moved down from lane 4 to the MUP. In this last step no time-lag will arise anymore, the pallets are moved down one by one and are transported out of the warehouse by the MUP.

This is a simple situation in which just a small amount of time-lag is created. In the real warehouses much more complex situations can arise in which pallets have to be dug-out on several levels and from behind lots of other pallets. The time-lag will than go up really quickly. And the time it takes to get out a requested selection of pallets goes up to an unacceptable amount.

Another possible situation in which time-lag arises is when the box is not sorted for some reason. This situation may occur when the box is too full to sort the warehouse in the available amount of time. Off course in this situation the warehouse is still able to access every pallet, but the time it takes to get the requested pallets out of the warehouse can go up really fast. Another common situation in which the box can be found in an unsorted condition, is when there are more different kind of products in the box as there are lanes. In this case the Optilogx box tries to sort every product on a separate lane, but can't find the solution for this problem. Therefore it will not reach the state of a totally sorted box. There are more situations possible in which the box is not able to get sorted to a perfect situation, but these situations described above are the most common ones. In these unsorted situations the box should still be able to provide the customers with the pallets requested, without the huge amount of time-lag that arises in these situations.

These situations are no exception in the real warehouses where the software of ORTEC is running right now. Therefore a lot of time-lag still arises when pallets are requested by the customers. In figure 2.39 a real-time situation at Hellendoorn is given in which the time-lag in the Optilogx box grew to an unacceptable amount. This is a situation which occurs frequently if the warehouse is relatively full. This includes many pallets inside the warehouse, but also the situation in which the amount of different products outnumbers the amount of lanes available inside the warehouse.

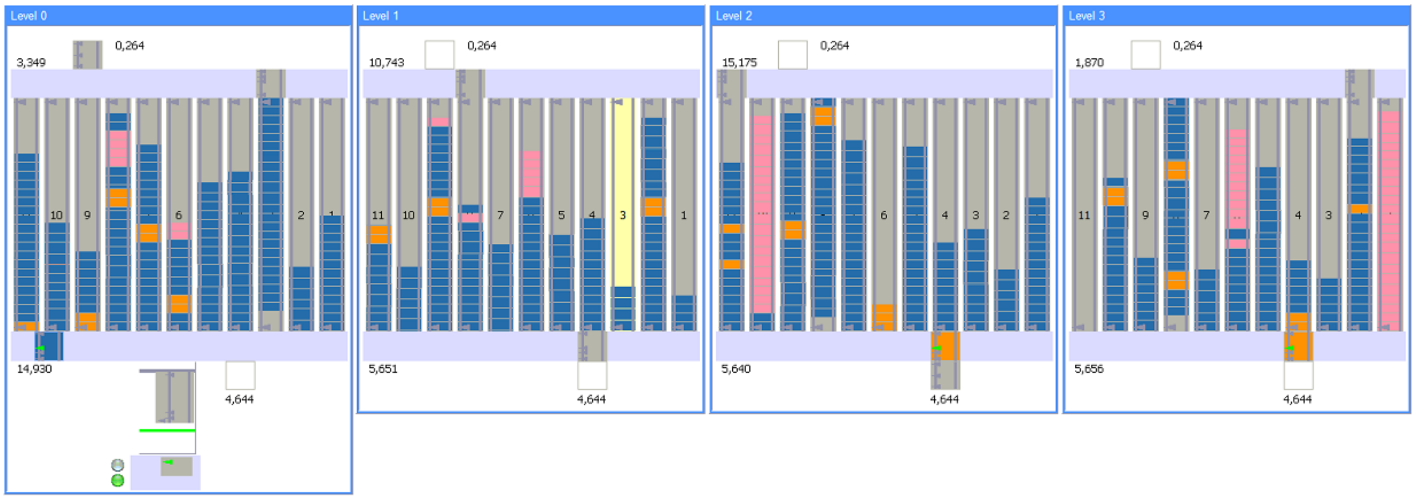


Figure 2.39: Unsorted warehouse

As described above, the orange pallets represent the pallets that are selected to go to the outfeed of the warehouse. The pink pallets are blocked pallets; these pallets are not allowed to leave the warehouse at this moment. In the figure allot of the selected pallets are located behind unselected pallets and therefore need to be dug-out. The situation above is a situation that occurs frequently in the warehouse at Hellendoorn. In this case the warehouse was not able to get sorted and therefore the selected order was spread all over the warehouse. This led to the situation in figure 2.39, which took more than 10 times as long as the technical capacity to move the pallets to the outfeed. On the other hand, it also took more than two times as long as the pallets could be retrieved if the warehouse was operated in the manual mode. In this manual mode the software does not generate the actions anymore, but the user gives the input to the system on which pallets have to move from where to where. The conclusion can be drawn that in some cases the operator can beat the OWCO software. Therefore in some cases the 'optimizer' is not really worth its name.

In order to prevent these kinds of problems, ORTEC is working on the feature called housekeeping. This process sorts every lane according to the different products inside the warehouse in order to achieve the ideal situation of a totally sorted warehouse. This housekeeping procedure works according to a number of steps that need to be carried out one by one. The housekeeping procedure starts when the warehouse does not have any important other moves to do; for example infeed or outfeed movements. It then starts with going through the housekeeping steps one by one to sort the warehouse into perfectly sorted situation. In the current situation the warehouse sometimes is too full for the housekeeping procedure to sort the entire warehouse. It will go through the first few steps but is not able to follow all the steps that need to be taken to get to a totally sorted warehouse. One of the reasons is that the warehouse is too full with pallets. Another reason is the amount of different products inside the warehouse. The housekeeping procedure tries to sort every product on a separate lane, and if the number of different products exceeds the number of lanes inside the warehouse, the ideal situation is impossible to reach. In this case when an order is requested, there is the possibility that (a part of) the order has to be dug-out, with all the time-lag problems involved.

The amount of time-lag that arises in these situations is in the most cases in the range of a 100% delay for the outfeed of one requested orderbatch. This means if an orderbatch normally takes half an hour to be moved to the outfeed, because of the time-lag it will now take 1 hour or even more to process the orderbatch request. This is an unacceptable amount of waiting time that the pallet-truck drivers and the 3PL truck drivers have to work with. The question arises: what can be done about this problem? This will be elaborated on in the 'analysis conclusion' in the next paragraph.

2.8 Conclusion analysis phase: research and sub-research questions

This paragraph describes the conclusions that can be drawn from the analysis phase described in this chapter. This chapter started with the comparison between the automated and non-automated warehouses. What kind of warehouses are commonly used and what is the specific functionality of these warehouses. The big advatages of an automated warehouse are the possibilities to increase the amount of pallets stored per square meter. Also the amount of human interaction with the pallets can be minimized. Because of labor and cooling costs, this is especially useful in warehouses operating in low tempuratures.

Other automated warehouses comparable with the Optilog box are the automated bicycle factory, the automated parking garages and the I-cube. From these, the most comparable with the Optilogx box is the I-cube.

The main components of the Optilogx box are described in paragraph 2.3 till 2.7. The technical capacity and the control software lay-out are described, along with the communication between these different software components and the processes inside the Optilogx box. The amount of time-lag that arises in some situations has a lot of influence on the procedures inside the Optilogx box. Especially the outfeed is severely affected by the situations in which (a lot of) time-lag arises in the box. The time-lag inside the Optilogx box can have several causes. The most important and most common situations are the cases in which pallets have to be dug-out. Another main problem in the warehouses is the fact that the optimizer does not have any awareness of resources or time. In the current situation actions are generated and are performed in the same sequence as they were generated (without some minor exceptions). In some cases this means that an action has to wait for another action to finish, while in the meantime the used resource could have performed another useful movement. An example is the usage of the VT, this resource is used by most of the jobs but only for a fraction of the entire job. When three jobs want to make use of the VT and have to be carried out right behind each other, the second and third actions have to wait for the VT to finish the first action. This creates a pallet 'queue' in the warehouse.

In order to prevent the first situation (digging out pallets), ORTEC is now working on the function of housekeeping that will make sure the warehouse is sorted as good as possible. This means the different products are preferably sorted in different lanes and these lanes are sorted by means of production dates (oldest product in front of the lane). However, this perfect situation is not always possible to achieve. An example is the already described situation in which the amount of different products outnumbers the lanes inside the warehouse. In this case some pallets of different products have to be placed on the same lane, leading to situation where time-lag in the box can arise. Also the number of colli on a pallet can differ, which can lead to the selection of pallets that are not in the ideal position. If such a situation occurs in the warehouse, all the pallets that are positioned in front of the selected pallets have to be moved to another lane. This again results in a situation with possible time-lag as a result. Therefore even in a perfectly sorted warehouse, the possibility of having to dig-out a pallet is always present with the result that time-lag still occurs regularly.

The goal of this research is to improve the performance of the total automated warehousing system. In order to achieve this goal, the focus of this research will be on minimizing the time-lag inside the Optilogx box of an automated warehouse. Therefore the following main research question was formed:

How can the technical capacity of an automated warehouse be reached by minimizing the time-lag caused by the Optilogx box, when focusing on the optimizer of the OWCO system?

In order to answer the main research question, sub-research questions are set up to gain more information about the different aspects of the main research question. Answering these sub-research questions will eventually lead to an answer for the main research question stated above.

1. ***Which existing logistical models can be used in the improvement of the optimizer's processes?***

2. ***How can the logistical models be implemented into a model, that will eventually lead to the minimization of the time-lag in the Optilogx box.***

3. ***What can be expected of the developed model? And how does the new model perform in comparison with the old model?***

This main research question with its sub-questions will be answered in the next chapters. Chapter 3 will focus on the design of the model, chapter 4 will present the results and performance of the developed model and finally the conclusions and recommendations are presented in the fifth chapter.

The design phase will start with elaborating on the logistical models described in sub-question 1. Different logistical models will be described and the comparison will be made with the problem present in the current OWCO software. The next paragraph will focus on the second sub-question and will present the model and software choice as well as a description of the goal of the model. In chapter 3 the model description will be given along with the mathematical formulation, the validation/verification and the placement of the newly developed model. Chapter 4 will focus on the results and the performance of the developed model and will therefore provide an answer to the third sub-question. The combination of the answers of the 3 sub-questions will eventually lead to the answer of the main research question in chapter 5.

3 Design Phase

Chapter 2 concluded the analysis phase with the main research question and three sub-questions. In this chapter an answer will be sought to the first two sub-questions. The first paragraph starts with discussing some logistical (types of) models with the focus on job- and flowshop scheduling. Then the model and software choice used in this research will be presented in paragraph 3.2 and 3.3. The link will be made between the developed model and the real-case scenarios of the Optilogx box at Hellendoorn, also the use of a solver will be presented. The mathematical formulation is presented in paragraph 3.4 followed by the implementation of the model in paragraph 3.5. Finally the verification and the validation of the model will be discussed in respectively paragraph 3.6 and 3.7.

3.1 Logistical models

3.1.1 Jobshop

The problem described in the conclusion analysis phase, shows some comparisons with a standard machine scheduling problem, for example a machine scheduling problem in a flexible manufacturing environment. Even in a very simple flexible manufacturing system (FMS) some input is required for the system to work properly. This input consists of product/machine restrictions; which product can be manufactured on what machine(s), as well as the sequence of the processes to manufacture the products on the various machines. Another important factor on these kinds of manufacturing processes is the sequence in which these processes are executed: which job is processed first and which ones thereafter? This sequence is the variable that can be changed to minimize the total amount of time necessary to process all the jobs.

These kind of problems are known as jobshop problems (Sviridenko, 2004) and consist of a number of lobs that have to be executed on a variety of machines. A job is a series of handlings that have to be performed on a number of machines. In a production environment this handling can be any process of adding value to the product on any machine. An example is adding the hood on a car in a car factory. Every job uses a machine for a certain amount of time and in this processtime the machine cannot be used to process any other jobs. In the most common formulation of the jobshop problem, every job has as many operations as there are machines: one job has to visit every machine. An important characteristic of the jobshop problem is the fact that the sequence in which a job visits the machines is not predefined (Alharkan, 1999). The next figure shows an example of a jobshop problem with an equal distribution of the process-times for every job on every machine. This leads to an optimized schedule with all the jobs 'touching' their subsequent jobs in the schedule. The horizontal axis represents time and the vertical axis represents the different machines used by the jobs.

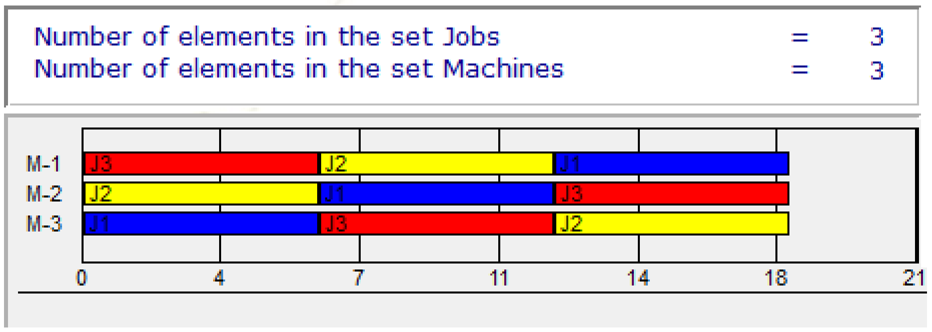

Number of elements in the set Jobs = 3
Number of elements in the set Machines = 3

Figure 3.1: Jobshop problem with an equal distribution

In the next example the same processes are shown in the same order on every machine, with the minor change in process-times for both Job-1 and Job-3 on every machine; these times are reduced with respectively 1/3 and 1/6. When looking at the finish time of the last machine, it can be concluded that the total time necessary to finish all the jobs is still equal to the previous example. This time necessary for completing every job, is called the timespan of the system. This example raises the general jobshop question: is it possible to change the sequence of the jobs on the machines to achieve a smaller total timespan.

The jobshop problem can be divided into two types of machine scheduling problems. The first is a scheduling problem in which the next job on a certain machine (for example Job-3 on Machine-3) can start, while the current job on that same machine (Job-1 on Machine-3) is not yet started on the next machine (Machine-2). This situation is displayed in figure 3.2, and is called a jobshop problem with pre-emption. In figure 3.3 the other situation is displayed: a jobshop problem without pre-emption: the next job on a machine can start if the current job is completed and started on another machine (Manikas & Chang, 2009). The difference between the figures 3.2 and 3.3 can be spotted in the starting times of Job-3 on both machine-2 and machine-3.

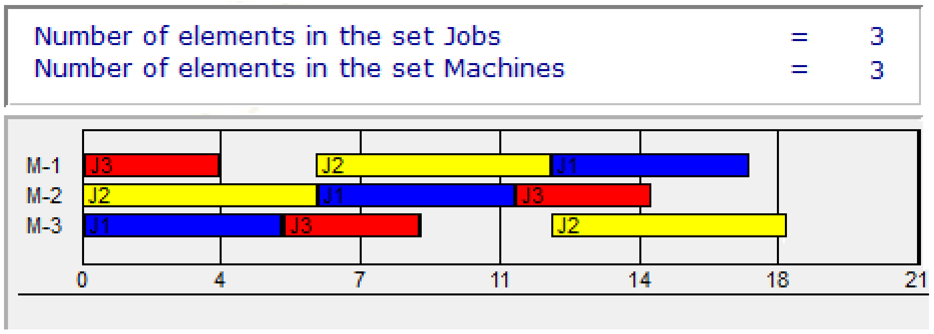

Number of elements in the set Jobs = 3
Number of elements in the set Machines = 3

Figure 3.2: Jobshop problem with pre-emption

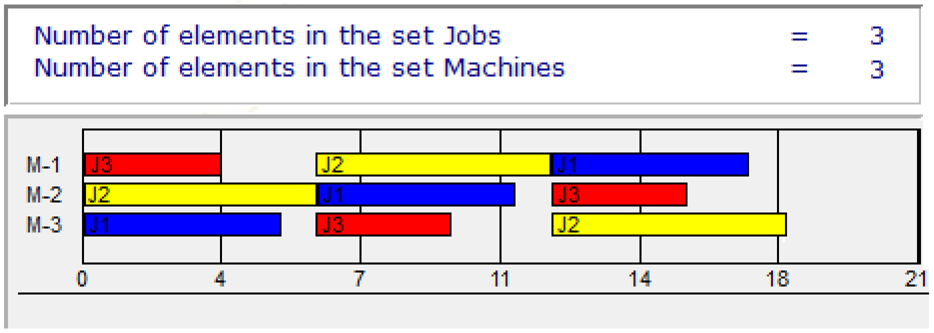

Number of elements in the set Jobs = 3
Number of elements in the set Machines = 3

Figure 3.3: Jobshop problem without pre-emption

3.1.2 Flowshop

There is a similar problem with a directional flow of the processes: the FlowShop problem. This problem is a special case of jobshop scheduling, in which there is a predefined sequence in the operations performed on the machines. This means every job has to visit **every** machine in a predefined order. An example of a flowshop problem is the permutation flowshop problem. This problem has a processing sequence of the machines which is similar for every job (Sviridenko, 2004). If the processing time for every machine is also the same for every job, the following figure can be constructed:

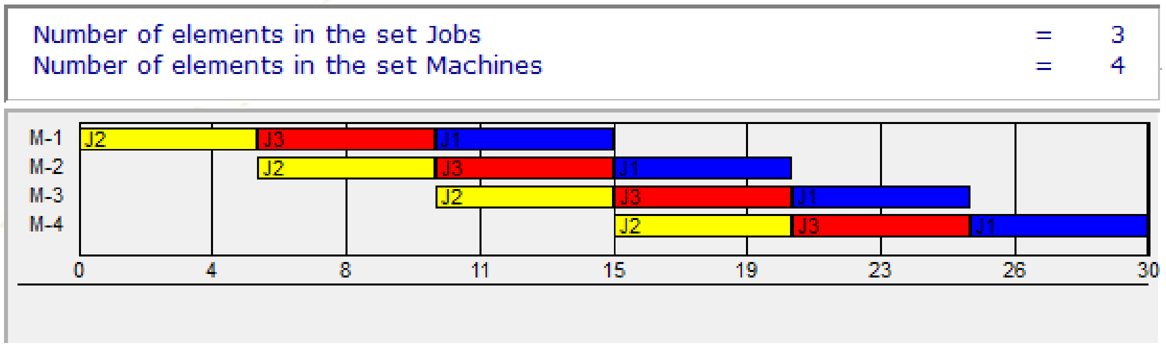

Number of elements in the set Jobs = 3
Number of elements in the set Machines = 4

Figure 3.4: Permutation flowshop problem

3.1.3 NP-hard/NP-complete

Both these problems (jobshop and flowshop) are closely related to the well-known Traveling Salesman Problem. This problem is defined by a group of cities (including the distances between these cities) and a salesman that has to visit all these cities exactly once to finally return to the city of origin. The problem of finding the shortest route in this situation is an NP-complete Problem. This means the problem is both in the set of NP problems (solvable in the non-deterministic polynomial time) and the set of NP-hard problems (Rock, 1984). The main characteristic of an NP-complete problem is the fact that there is no known efficient way to find a solution for these problems. This means there is no algorithm available to determine a fast solution. Also the time required to solve the problem increases very fast as the size of the problem grows (Papadimitriou & Kanellakis, 1980).

Another well-known problem in the category of NP-complete, is (a variation of) the vehicle routing problem or VRP. This problem is an optimization problem in which a number of customers have to be serviced by a certain amount of vehicles. (Dantzig & Ramzer, 1959). This VRP is one of the more important problems in the field of transportation, distribution and logistics. This problem is comparable with the Traveling Salesman problem in the way that they are both NP-Complete problems.

Like the traveling salesman and the VRP problem, the (permutation) flowshop problem is NP-hard/complete if there are m = 3 machines or more. Logically the jobshop problem (which is even harder to solve) is also NP-hard/complete. This was already proven in 1976 by Garey (1976). With this prove the conclusion can be drawn there is no standard available algorithm to solve these kinds of problems, therefore when developing a model to solve a job/flowshop problem, there is an important decision to be made on how to come to the desirable result. This will be discussed in the next paragraph.

3.2 Software choice and the goal of the model

When looking at the resources inside the warehouse at Hellendoorn, the situation of outfeed shows resemblance with the situation of a jobshop and flowshop problem. The machines in a job/flowshop problem can be linked to the resources inside the warehouse. Every resource will get its own machine number inside the Optilogx box. In this case the machines will not add value to the product, but will provide the transportation of a pallet from A to B. A job can therefore be described as the total action of moving a pallet from location A to D on pallet level, whereby every resource/machine takes care of 1 pallet movement in the entire job.

The pallet movements will make use of a selection of machines; not every pallet moved to the outfeed will make use of every machine. Therefore the problem is not similar to a standard or permutation flowshop problem, but also shows some aspects of a jobshop problem without pre-emption.

When focusing on the outfeed side of the warehouse, there are 6 machines/resources involved in transporting the pallets to the outfeed (4 shuttles, the VT and the MUP). The lanes from which a pallet will be retrieved will be assumed to be positioned at the outfeed side as well. Therefore the lanes can be excluded from the outfeed process, and only these 6 machines will be involved: The first machine is the shuttle at the top level (Level-3), which is connected to the Vertical Transporter (VT). The same connection is established for the shuttles on the levels 2 and 1. The VT is then connected to the shuttle on level 0, which finally is connected to the MUP. The main difference with a normal flowshop problem is the fact that the shuttles on the levels 3, 2, 1 and 0 are all connected to the same resource: the VT.

In this situation some jobs will skip machines, and others will only make use of the last 2 machines. This combination of using and skipping machines, leads to the outfeed situation at the Hellendoorn warehouse which is comparable with a permutation flowshop problem in combination with a jobshop problem without pre-emption. Because there is no standard solution algorithm to solve the separate job/flowshop problems (see also paragraph 3.1.3), there is also no solution algorithm available for this combination of the two problems.

This combination of flow and jobshop problems both have integer values in its formulation, as well as non-integer values (for example a binary value in the model, which indicates whether or not a job is assigned to a specific position in the schedule). Therefore this problem can be described as a Mixed Integer Problem (MIP). The formulation of the MIP problem will be further discussed in the next chapter.

In order to make a distinction between the machines, every machine will get its own number. The numbers are assigned to the machines by following the path of the pallets down to the outfeed from the top level inside the warehouse. This will be the resource (shuttle) at the top level: Machine-1. The shuttle on level-2 will then be the next machine: 2 and the shuttle on level 1 will be machine-3. The VT connected to all these previous machines will be machine-4. The next connection will be the shuttle on level-0: machine 5, and finally the MUP will be the last machine: 6. Figure 3.5 shows the resources inside the warehouse with its machine-names.

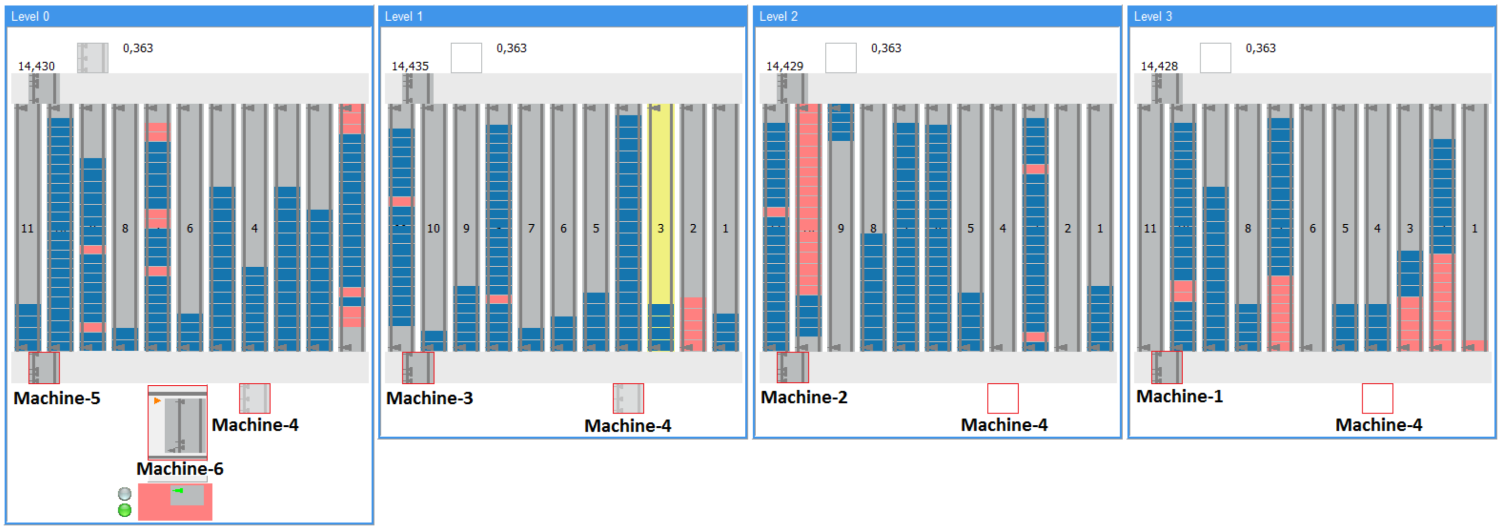


Figure 3.5: Machine numbers

3.2.1 Developed Model (the goal)

Paragraph 2.7 describes the main problems inside the current warehouse; the situation in which pallets have to be dug-out and the fact that the optimizer does not have any awareness of resources or time. The second problem leads to the situation where resources are waiting on actions to finish, while they could have performed another useful movement. For example the VT situation described in paragraph 2.7, in which a pallet 'queue' arises in the warehouse.

There could be a solution found for every specific situation, but it would be more preferable if one solution was found to minimize all these problems. This solution could be to develop a scheduler/planner for the optimizer that takes both time and resources into account. In the current situation actions are generated (see also paragraph 2.5) and are performed in the same sequence as they were generated (without some minor exceptions). The scheduler would optimize this sequence in which the actions are performed by the control software. This will limit the amount of time-lag to a minimum. The scheduler will be developed for the Optilogx box, mainly because the Optilogx box can be seen as the heart of the total system: every action of in- and outfeed is moved in or through the Optilogx box. Therefore often the most time-lag is present inside the box. With this scheduler the time-lag will be minimized.

The model environment 'AIMMS' is used to develop a model (see also paragraph 3.3) which makes it possible to schedule all the separate pallet actions in such a way that the total timespan can be calculated. This is done by building a model consisting of sets, parameters, variables and constraints. The objective function of the model is to minimize the variable of the total timespan, while taking the values of the parameters and sets as starting point and satisfying the conditions of the constraints. The goal of the model therefore can be described as: optimizing the sequence in which the pallets will be moved to the outfeed, with the goal of scheduling the last action as early as possible in the time and therefore minimizing the total timespan. The input for the model will be a set of actions that represent an outfeed orderbatch. This set of outfeed actions is built up from a routing table as can be found in appendix B. This input will be picked up by AIMMS from an Excel file. The model will then optimize the sequence in which the actions are executed inside the warehouse, to finally present this pallet sequence as the outfeed of the model. A sum-up of this process can be seen in figure 3.6.

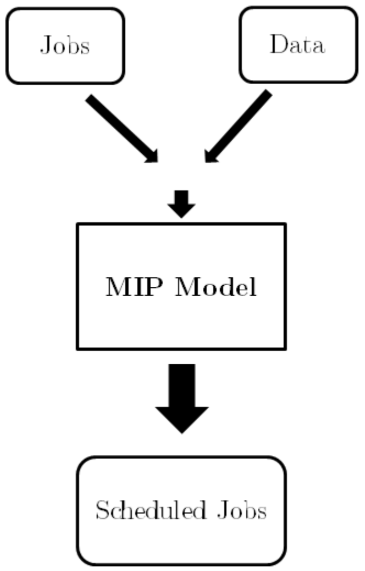


Figure 3.6: Infeed and outfeed of the MIP Model

To visualize the outfeed of the model in an organized way, the use of Ganttcharts is deliberately chosen. Ganttcharts are easy to understand, and can give a clear and colourful visualisation of the problem. The model will then be verified and validated, and the outfeed of the model (a pallet sequence) will be tested with real case scenarios. These test results will be compared with the real cases from the warehouse at Hellendoorn.

A model is a simplification of the reality; therefore the resource times used for the calculations in the model should be as close to the reality as possible. The way to achieve this is to determine the times all the different resources need to process a pallet. This will be done by calculations and measurements (see appendix C). The determined values will then be rounded to whole seconds. Some other assumptions are made for the model, first of all the processtimes on a resource is similar every time this resource is used. The pallets can only be moved to the outfeed from the outfeed side of the warehouse. This leads to the situation in which only one route is possible for a lane to the MUP. This assumption is similar to the configuration of the OWCO software at this moment in Hellendoorn. For the model it can be assumed that a dig-out action can always find a new pallet place at the same level as it was originally positioned. This means only a shuttle is needed to carry out a dig-out action. The lanes from which the pallets are moved to the outfeed are assumed to be positioned at the outfeed side of the warehouse. The final assumption is done on the pallet size inside the warehouse. For the selected pallets for an orderbatch, the pallets all consist of EURO pallets.

3.3 Design of the model

3.3.1 The model description

The model is constructed by making use of different elements. First different sets are defined in the modelling environment. These sets include the different elements used in the other segments of the program, for example in the different parameters. These parameters consist of the data included in the different situations of the problem. Examples of these parameters are the times necessary for a job on a certain machine. This information can be changed for other warehouses that can be calculated in the model. The next elements of the model are the different variables used by the solver in order to determine a solution. The variables can consist of the starttimes of every action on every machine, the timespan of the total procedure or for example the position of outfeed. The variables have to fulfil certain conditions, which are formulated as constraints inside the model. Examples of some constraints are the restrictions on the number of jobs per machine or a restriction on the starttime of certain machines. These elements are all used by a mathematical program which invokes the solver to start working towards a solution by making use of the different variables and taking the constraints into account. The goal of the developed model is to minimize the variable total timespan, for a certain group of actions. This total timespan can be explained as the time at which the last action on the last machine takes place. The total timespan is equal to the total time needed to perform all the actions.

The infeed for this model is a group of jobs/actions; these actions represent the movements of the pallets inside the warehouse. The focus of the model is on solving the outfeed situation. Therefore the actions that are used as infeed for the model are all related to the outfeed procedure. The actions are made by top layer of the ORTEC software: WPMS. The model does not delete or (re)create any new actions. The actions consist of a location from which a pallet is moved, connected to the location where the pallet is moved to. An example of an action is: L3-01 → L3-02. This action will move the first pallet on level-3 lane 1 to level-3 lane 2. All these actions will also include the resources they use, and the time they keep this resource occupied. All this information will be imported from Excel and is available in AIMMS as the starting situation. The outfeed of the model is a Ganttchart with a visual representation of all the actions performed on the different machines and a matrix in which the sequence of the outfeed actions is presented. This matrix can be used to compare different situations in the emulation tool available for the Optilogx box.

The time it takes for a pallet to fulfil a certain actions on a certain resource in the model, should be equal to the times in the real situation. In order to determine these times, some calculations and measurements have been carried out at the Hellendoorn warehouse. These calculations and measurements can be found in appendix C, and are used in the model as the travel times. The transfers of a pallet between the different resources can be seen in the model as a pallet that is located at both of the resources involved in the transfer. Therefore both the resources are busy for a time of 8 seconds. This can be seen as overlapping time-blocks in the Ganttchart of the model. The movements (Job-1 from VT to the MUP) are displayed in blue, and the transfers in lighter blue. It is clear from this figure that two resources are busy with the same Job at the same time.

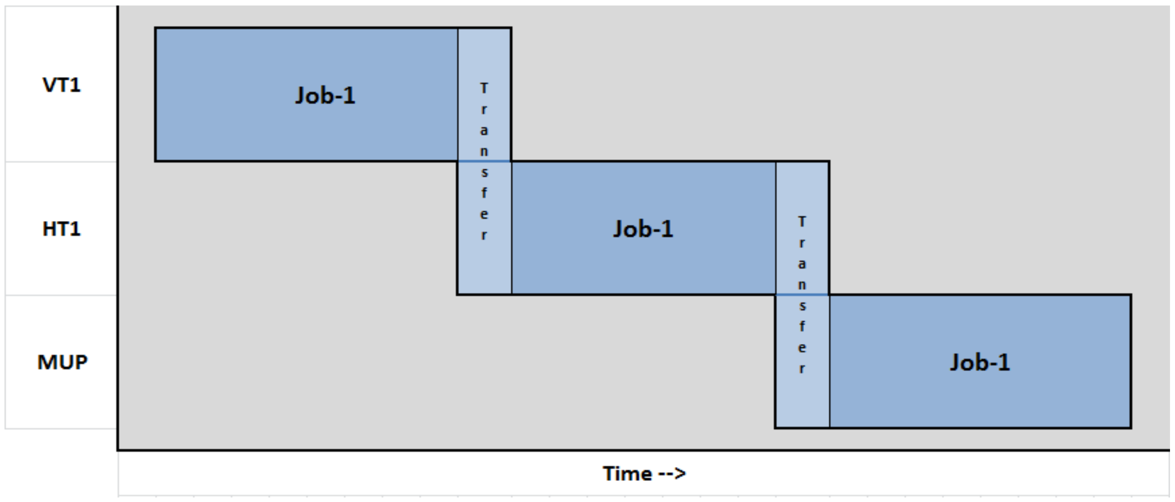


Figure 3.7: Overlapping time-blocks

The group of actions that is provided as the infeed for the model represents an outfeed orderbatch of a full truckload. This is a standard situation in the case of the Hellendoorn warehouse. The model calculates every separate outfeed orderbatch, which consist of all the movements and transfers necessary to move the selected pallets to the outfeed. To make sure the resources take the same amount of time in the model as they do in reality, some calculations are done to determine these resource-times, ride-times and transfers. Also some measurements have been done at the warehouse to determine the resource-, ride- and transfer-times. Both these procedures and the average measured results are given in the appendix C. The resource-, ride- and transfer-times are then used to build up an Excel file with all the actions and the times necessary on the resources. The model will pick up all the data from the Excel file and use this to determine the new outfeed sequence of pallets. This method of communicating all the (resource) data from Excel, is deliberately chosen to keep the model as generic as possible. If another resource is used inside this warehouse or the entire lay-out of the warehouse changes, this model can be used with only a small amount of changes. This is an important characteristic of the model for its generic use in different warehouses.

3.3.2 Solver

In order to find a feasible solution in the model, a tool was needed to construct and solve the model. For this tool a solver was chosen and the environment this solver is used in. For the environment, the program AIMMS is chosen. The reason for this is the large amount of well-rated solvers that are available in AIMMS and the compatibility of AIMMS with other programs like Excel. AIMMS is commonly used for the development of comparable (MIP) models and to tackle other related mathematical optimization problems (non-linear, e.g.). Therefore AIMMS proved to be the best practical means for the development of this specific model. Another big advantage of AIMMS is the possibility to construct a clear visual presentation of the outcomes of the model.

The different solvers available in AIMMS could be tested in order to find a suitable one in the model environment. Not every solver can be used for each problem; in this case the focus will be on solvers that can handle a MIP problem. From the literature[19], two solvers available in AIMMS are far better than the others. Therefore the choice has to be made between these different two solvers: CPLEX 12.5 and GUROBI 5.1. The benchmark tests show comparable calculation time results, but the GUROBI solver tends to have more peak values in the calculation times of complex situations. This can also be seen in the log files[20] of the benchmark test. Therefore the decision is made to make use of the CPLEX 12.5 solver to test the developed model.

This solver makes use of a branch-and-bound algorithm in combination with cutting planes and heuristics to find integer solutions. In combination with the pre-solver, this solver proved to be a good practical means in order to solve this problem. A big advantage of the CPLEX solver is also the fact that it is possible to decouple the solver from AIMMS. The solver can then be used as an individual program, and can be linked to another program. This brings the possibility to use the solver in the existing warehousing software like CP or WPMS. The disadvantage of this solver is the (high) license cost that the use of this solver brings.

3.4 Mathematical formulation

Paragraph 3.2 and 3.3 give a description of the developed model. Taking this description as a starting point, the mathematical formulation of the MIP model will be presented. First the parameters and sets will be described and summed up prior to the explanation of the model and the objective function. Finally the actual formulas will be presented. The assumptions made with respect to this model are described in paragraph 3.2.

The actions necessary to complete the outfeed of the orderbatch are taken as the starting point of the model. Every pallet that is moved has its own separate action. This action consists of the current position of the pallet and the position it will be moved to. These actions/pallet movements will be described as the jobs inside the model. Suppose there are N *Jobs*, indexed by j = 1,..,N and $\mathcal{J} = \{1,...,N\}$. The model described will plan the jobs into a time horizon, whereby the sequence of the jobs in which they will be performed, is defined by *position* P, p = 1,..,P and $\mathcal{P} = \{1,...,P\}$. Hereby the number of jobs is equal to the number of positions. The resources can be defined as the *machines* M inside the model: m = 1,..,M and $\mathcal{M} = \{1,...,M\}$. In this case starting with the resource at the top level: L3-HTx51 as m = 1, then L2-HTx51, L1-HTx51, VT-x51, L0-

[19] http://plato.asu.edu/ftp/feas_bench.html

[20] http://plato.asu.edu/ftp/feas_bench_logs/

HTx51, and finally the MUP as M (m = 5). This can also be seen in figure 3.5: Machine numbers. In order to be able divide the pallets into different groups, *JobGroups* are indexed by g = 1,..,G and $\mathcal{G}$ = {1,...,G}. These jobgroups will be used in order to specify a group of pallets that will be dug-out in the warehouse.

Jobs	j	=	1,..,N
Positions	p	=	1,..,P
Machines	m	=	1,..,M
JobGroups	g	=	1,..,G

The P*rocesstimes* of each job is given by $PT_{j,m}$ for which it holds that each job has a processtime for every machine. This time will be equal or larger than 0. When no processtime is formulated for a resource (the job is not using the resource), $PT_{j,m}$ will automatically take the value 0. In order to link a starttime to every action, the variable *StartTimes* is formulated. For which it holds that $ST_{p,m} \geq 0$ for j = 1,..,N. *Jobpositionschedule* is the decision variable to determining the position at which each individual pallet will be moved relative to the other jobs. This $JPS_{j,p}$ is a matrix consisting of every job and every position, for which it holds that every Job can be linked to only one position and every position can only be linked to one job.

The time it takes for a pallet to be transferred from one resource to another is given as the *TransferTime* TT in the model. This transfertime is the time at which both the resources involved in the transfer are busy with the same job. In order to define the machine number which the vertical transporter has, VTMnr is defined as the *Vertical Transport Machine number.* This number of the machine will be used to define the pallet flow from the machines 1, 2 and 3 (the shuttles on level 1 till 3).

Every job will start from a certain location inside the Optilogx box. This start location will be given by the *JobLocation* $JL_{j,l}$. Lets define the *JobGroupMatrix as* $JGM_{j,g}$. This matrix gives the JobGroups that are assigned to every Job, in which the selected pallets are in a different group as the pallets that have to be dug-out and the machine which the pallet uses for the first movement.

$PT_{j,m}$	=	*ProcessTime*
$ST_{p,m}$	=	*StartTime*
$JPS_{j,p}$	=	*JobPositionSchedule*
TT	=	*TransferTime*
VTMnr	=	*Vertical Tranporter Machine number*
$JL_{j,l}$	=	*JobLocation*
$JGM_{j,g}$	=	*JobGroupMatrix*

The objective function of the model is to minimize the total timespan: this timespan is the total amount of time it takes to process all (outfeed) jobs inside the Optilogx box. To determine the total timespan, the last action on the last machine is taken and the duration of this action is added to the starttime of the action. The objective function will minimize the total timespan and is given in formula (1).

The requirement to this objective function is the condition of certain constraints that have to be satisfied. In order to guarantee only one job is linked to one position, the summation is taken over the jobs in the JobPositionSchedule matrix. This summation has to be equal to 1 for every job (2). In this same way the constraint is given to guarantee only one position per job, by taking the summation over the positions for every job (3).

The next step is to define the starttimes of the various machines and jobs in the model. First the starttime of the next job on a certain machine will be given. This constraint will make sure the next job on the same machine will not start before the current job has ended. This is done by stating that the starttime of the next job is larger or equal to the starttime of the current job + the duration of this job (4). In this way the next job can only start if the current job is finished.

To guarantee the correct starttime of the same job on the next machine, it is stated that the starttime of the same job on the next machine has to be larger or equal to the starttime of the job on the current machine + the duration of this job – the transfer time (5). This will give an overlap of the transfer time. This constraint is valid for the resources that have a machine number higher or equal to the number of the vertical transporter. This because of the 'jump' a pallet can make from the machines 1, 2 and 3 to the VT (machine 4 in this case). In order to make the correct transfer possible from the first 3 machines to the VT, it is stated that the starttime of the same job on the next machine is given by the starttime of the job on the current machine + the duration of this job – the transfer time (6). For this constraint the next machine can be defined as the current machine + i, in which i is the VT machine number – the number of the current machine. This will give the number of 'steps' the pallet has to make in order to arrive at his next machine (the VT).

In order to make sure the dig-out actions take place before the selected pallet(s) behind these 'dig-out pallets', it is stated that a pallet of the next jobgroup on the same startlocation and using the same first resource, has to start after the pallets in the current jobgroup (7). This will guarantee the selected pallets to be only moved down if all the pallets of a previous jobgroup ('dig-out pallets' in front of the selected ones) are removed.

Objective Function:

$$Minimize \qquad ST_{N,M} + \sum_{j \in N} (PT_j^N * JPS_{j,p}) \tag{1}$$

s.t.

$$\sum_{j \in J} \mathrm{JPS_{j,p}} = 1 \qquad \forall\, p \in \mathcal{P} \tag{2}$$

$$\sum_{p \in P} \mathrm{JPS_{j,p}} = 1 \qquad \forall\, j \in \mathcal{J} \tag{3}$$

$$\mathrm{ST_{(p+1,m)}} \geq \mathrm{ST_{p,m}} + \sum (\mathrm{PT_{j,m}} * \mathrm{JPS_{j,p}}) \qquad \forall\, m \in \mathcal{M}, p \in \mathcal{P} \tag{4}$$

$$\mathrm{ST_{(p,m+1)}} \geq \mathrm{ST_{p,m}} + \sum (\mathrm{PT_{j,m}} * \mathrm{JPS_{j,p}}) - \mathrm{TT} \qquad \forall\, m \geq VTMnr \tag{5}$$

$$\mathrm{ST_{(p,m+i)}} \geq \mathrm{ST_{p,m}} + \sum (\mathrm{PT_{j,m}} * \mathrm{JPS_{j,p}}) - \mathrm{TT} \qquad \forall\, m < VTMnr, i = VTMnr - m \tag{6}$$

$$\mathrm{JGM_{j,g+1}} \geq \mathrm{JGM_{j,g}} \qquad \forall\, g \in \mathcal{G} \tag{7}$$

With:

$\mathrm{PT_{j,m}}$	=	*ProcessTime*	*Jobs*	j	=	1,..,N
$\mathrm{ST_{p,m}}$	=	*StartTime*	*Positions*	p	=	1,..,P
$\mathrm{JPS_{j,p}}$	=	*JobPositionSchedule*	*Machines*	m	=	1,..,M
TT	=	*TransferTime*	*JobGroups*	g	=	1,..,G
VTMnr	=	*Vertical Tranporter Machine number*				
$\mathrm{JL_{j,l}}$	=	*JobLocation*				
$\mathrm{JGM_{j,g}}$	=	*JobGroupMatrix*				

3.5 Implementation of the model

Because of the limited amount of time available, it was not possible to build the scheduler into the software of ORTEC. The scheduler was developed as a standalone tool. However, the place of the scheduler is an important point of attention. Therefore this paragraph describes the position where this model could be placed inside the ORTEC software.

The model described in the previous paragraph can be seen as a separate tool to schedule all the outfeed actions of a requested orderbatch. This tool is now a standalone component, which uses the pallet movements as input. The output can be rearranged as a list of the sequence in which the pallets have to be moved to the outfeed. This pallet sequence is the input for CP that should be used when executing the actions. To determine an implementation location for the developed scheduler, figures 3.8 till 3.11 show the proposed positions displayed in the simplified software diagram. The scheduler is presented in orange and has to be positioned somewhere it can receive the actions generated by WPMS. The scheduler will then reply with the new sequence of these actions. The first possible placement is given in figure: 3.8, in which the scheduler is placed inside CP. In this case the CP software will receive the pallet actions from WPMS and will reschedule these actions to a better sequence. The second possibility is to place the scheduler between CP and WPMS. In this case WPMS will send the actions to the scheduler, and then the scheduler will send the actions to CP. In the third figure, the scheduler is placed inside the WPMS software. Before sending the action towards CP, WPMS will put the action in a new sequence by making use of the scheduler. The last possibility is to place the scheduler alongside WPMS. This means before sending the actions to CP, WPMS will first send them to the scheduler.

However, there are some disadvantages to the positions of the scheduler. The disadvantage of the first position is the fact that CP is merely used for the direct control of the warehouse and should not be used for anything else. The main disadvantage of both the second and the fourth location is the necessity of having extra communication between the different software components and the scheduler. Therefore after some discussions, the decision was made to place the scheduler inside WPMS (figure 3.10). Because of the limited amount of time available for this project, it was not possible to build the scheduler into the OWCO software. Therefore only a recommendation can be done to position the scheduler inside WPMS.

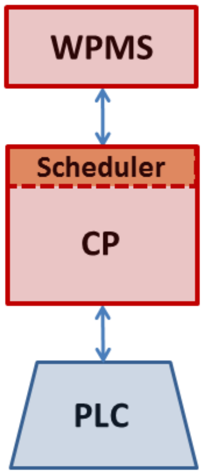


Figure 3.8: Scheduler inside CP

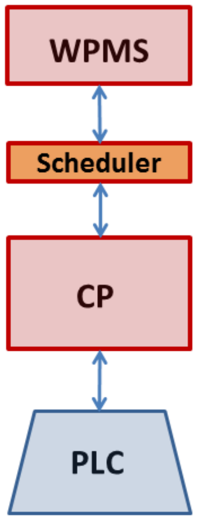


Figure 3.9: Scheduler between CP and WPMS

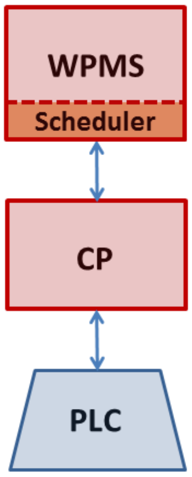


Figure 3.10: Scheduler inside WPMS

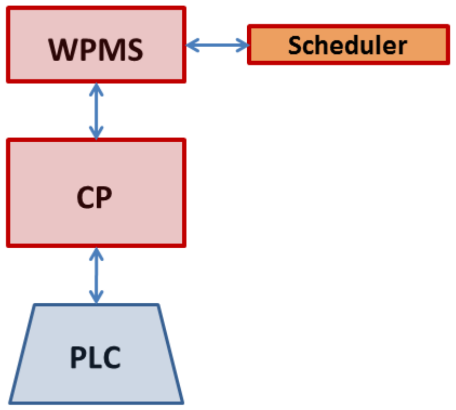


Figure 3.11: Scheduler alongside WPMS

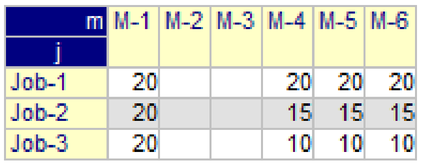
3.6 Verification of the model

Before the model can be implemented, a verification and validation has to be performed. This will be done according to the paper on “model verification and validation” by Sargent (2004). This paper describes the model verification as: “ensuring that the computer programs, the computerized model and its implementations are correct”. Therefore the focus will be on verifying whether or not the model is correct. In order to verify the model, the verification by hand-simulation technique is used. This common software verification technique compares (simple) hand calculations with the outcomes generated by the model. The comparisons are made between four situations in which all the main functionalities of the model are present. These cases are made-up situations in which the processtimes of the resources are customized. These resource times can deviate from the real situation at the Hellendoorn warehouse.

The developed model is a deterministic model, in which only the optimal outcome will be presented. This solution is optimal, given the boundary conditions and the assumptions made for this model. If the optimal sequence is not found for whatever reason, no answer will be presented. The model can be seen as verified when all of the presented comparisons provide outcomes which are similar to the minimal timespan determined by the hand calculation. In paragraph 3.7 the validation of the model will be discussed according to the same paper on “model verification and validation” (Sargent, 2004).

3.6.1 Situation 1

The first small verification situation is the condition in which three pallets are moved down from L3 (M-1) to the MUP (M-6). The model should place these actions behind each other while taking the time these actions take on the separate resources into account. In figure 3.12 the matrix is given in which the jobs and the machines can be seen. The machines can be seen in the upper horizontal row and the jobs are given in the first column of the matrix. The jobs all start at machine 1 and end on machine 6. In these situations the overlap time between the different resources is set to 4 seconds. In order to determine the total timespan a hand calculation can be done. This calculation can be seen next to figure 3.12 and represents the minimal time needed for all the actions to finish. First Job-1 on M-1 is carried out (20 sec), then this same job is carried out on M-4, taking the overlap time into account (20 - 4 sec). This same job is then carried out on the machines 5 and 6, again taking the overlap times into account (20 - 4). Job-1 is finished, but job-2 and 3 are not. Because in this customized example the jobtimes of job-1 on machine 4, 5 and 6 are larger than the jobtimes of job-2 and 3 on the machines 4, 5 and 6. Only the last machine then still needs to process job-2 and job-3. This will take 15 and 10 seconds respectively.



m	M-1	M-2	M-3	M-4	M-5	M-6
j						
Job-1	20			20	20	20
Job-2	20			15	15	15
Job-3	20			10	10	10

$$20 + (20 - 4) * 3 + 15 + 10 = 93$$

Figure 3.12: Matrix situation 1

The Ganttchart is then constructed by the model and given in figure 3.13 This Ganttchart shows a total timespan of 93 seconds, similar to the hand-calculations done in prior to the model calculation. The similarity of these values indicates the model found the optimal solution.

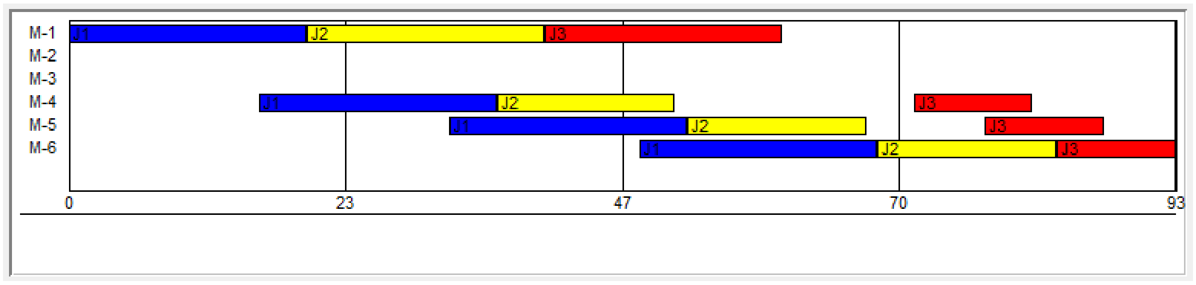
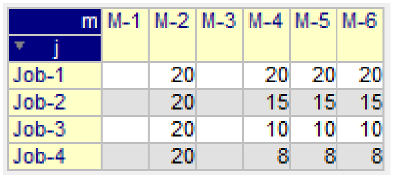


Figure 3.13: Ganttchart situation 1

3.6.2 Situation 2

The second situation represents a case in which 4 pallets starting from L-2 (m-2) will be moved to the MUP (m-6). This case has a fourth pallet with jobtimes smaller than the previous jobs. Therefore this fourth pallet should elongate the total timespan with the processtime on M-6 (8 sec). These processtimes are made-up for this verification example and are not representative for the real warehouse at Hellendoorn. Again the calculation can be made to determine the smallest timespan:



m / j	M-1	M-2	M-3	M-4	M-5	M-6
Job-1		20		20	20	20
Job-2		20		15	15	15
Job-3		20		10	10	10
Job-4		20		8	8	8

$$20 + (20 - 4) * 3 + 15 + 10 + 8 = 101$$

Figure 3.14: Matrix situation 2

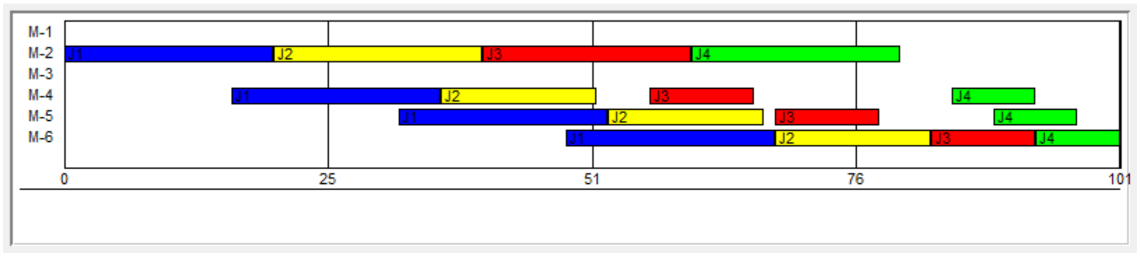
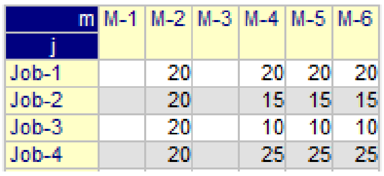


Figure 3.15: Ganttchart situation 2

The Ganttchart shows the total timespan to move all the pallets to the outfeed. No time-lag is present in this situation. The total timespan (101) is similar to the pre-calculated value and thus the optimal solution was found by the model.

3.6.3 Situation 3

Situation 3 presents a case in which again four jobs are scheduled. The difference is the processtimes of the fourth job. These processtimes are bigger than the processtimes of the previous jobs. Therefore this fourth job is expected to be positioned as the first job in the sequence of 4 in order to come to the optimized timespan. Again a hand calculation provides the minimal total timespan:



m / j	M-1	M-2	M-3	M-4	M-5	M-6
Job-1		20		20	20	20
Job-2		20		15	15	15
Job-3		20		10	10	10
Job-4		20		25	25	25

$$25 + (25 - 4) * 3 + 20 + 15 + 10 = 128$$

Figure 3.16: Matrix situation 3

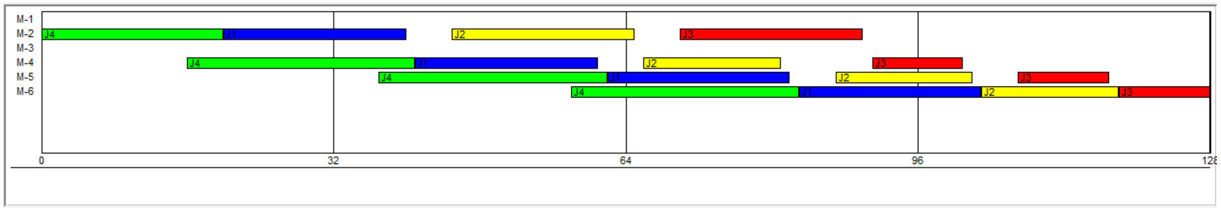
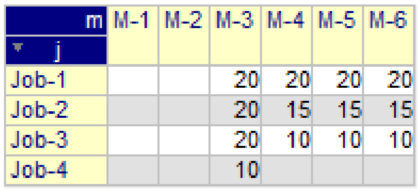


Figure 3.17: Ganttchart situation 3

The Ganttchart presents a solution of 128 seconds, which is equal to the hand calculation. Therefore it can be said the model has found the optimal solution to this problem as well.

3.6.4 Situation 4

The fourth situation describes a case in which a pallet needs to be dug-out. The dig-out pallet is positioned on L1 (m-3) and should take place before the jobs 1, 2 and 3. The total timespan should in this case be 103 seconds.



m / j	M-1	M-2	M-3	M-4	M-5	M-6
Job-1			20	20	20	20
Job-2			20	15	15	15
Job-3			20	10	10	10
Job-4			10			

$$10 + 20 + (20 - 4) * 3 + 15 + 10 = 103$$

Figure 3.18: Matrix situation 4

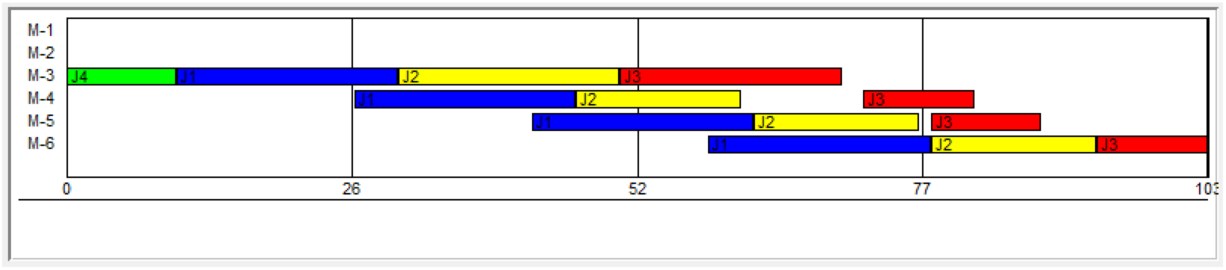


Figure 3.19: Ganttchart situation 4

As can be seen in figure 3.19, the model constructs the timespan without any time-lag in the Ganttchart with a total timespan of 103 seconds, equal to the predefined minimum timespan. Also the pallet that needs to be dug-out before the other pallet movements can take place, is correctly moved as the first action (see the green J4 in figure 3.19).

The developed model is a deterministic model, in which only the optimal outcome will be presented. Because the model is deterministic, every run will provide the same minimized timespan. Therefore it is not necessary to run the different situations multiple times. Every run will again provide the exact same answer: the minimal timespan which is equal to the optimized solution. The only difference when running the same situation multiple times could be the calculation time of the model.

In all these tested situations, the developed model provides the optimal solution for the given condition. This optimal solution is equal to the value found in the hand-calculation done in advance and therefore the model can be seen as verified.

3.7 Validation of the model

The model validation is defined as: “substantiation that a computerized model within its domain of applicability possesses a satisfactory range of accuracy consistent with the intended application of the model” (Schlesinger, 1979). This validation will focus on determining whether or not it is the correct model for providing the answers regarding the real world. It is not always possible to do a validation, because in many cases the validation only can be done when implementing the model into real-world scenarios. In this specific case there is an emulation tool available developed by ORTEC which is an exact copy of the real warehouse situation. Every component takes exactly the processtime as it will take in the real warehouse. After extensive testing, the assumption can be made that the emulation tool represents the real warehouse situation. This tool can be used as a first step towards the validation of the model. The actual validation will be done when the new model will be implemented into the software of the Hellendoorn warehouse.

In this stage a first step towards the validation of the developed model can be carried out. This first step will focus on validating the answers provided by the model regarding the real world. The emulation tool developed by ORTEC to simulate the different warehouses is used. This emulation tool uses the exact same software as the customers have installed for the control of their warehouse. Therefore the same calculations are behind the emulation tool, meaning every component takes exact as much time for an action as it does in reality. The processtimes of the resources can be retrieved and used as the input values for the model. The values of both outfeed actions and dig-out actions can be seen in the appendix B.

For the validation of the model, the comparison will be made between the timespan generated by the model, and the time it takes to actually move all the pallets to the outfeed when making use of the emulation software. These total timespans will be compared and from these values a conclusion can be made towards this first validation step of the model. The test results described in the next chapter are used for the validation.

In the cases described in the next chapter, the total timespan determined by the model is compared with the timespan generated when putting the optimized sequence of the cases into the emulation tool. These values are represented by the MIP Model and the Emulation MIP presented in the test results. The comparison of these values leads to the conclusion that the values match in almost every situation. No more deviation than 1 second was found. This is a logical finding, because the same assumptions are done for the model and the emulation tool. This first step towards the validation of the model can therefore be used for the validation of the deviation errors inside the model. This deviation can be explained by the fact that the new model only uses round-up values as the input for the resource times, while the emulation tool uses values accurate to the millisecond. With these test results, the first step of the validation process can be seen as a success. Therefore the values determined by the model when implementing this into the real warehouse are expected to give the same results as in these test cases. However, the actual validation of the model can only be done after the implementation at the Hellendoorn warehouse.

4 Model results and performance

Chapter 3 describes the newly developed model, including the mathematical formulation, verification and validation of the model. In this chapter, the model will be tested by making use of several scenarios and finally the results will be presented and discussed. In the next chapter the conclusions will be presented with the recommendations and possibilities for further research.

4.1 Cases: the performance of the model

As described in the previous chapter, the outfeed of the model will be a sequence in which the pallet actions are executed. In this chapter some cases will be presented to test the model's performance. How well does the scheduler work in comparison with the old situation? This will be determined by making use of several cases of outfeed situations in the automated warehouse. In the real warehouse the outfeed situation is dependent on the situation inside the box. In a relatively empty box, the pallets have a bigger chance of being positioned at the front of the lanes. However in a relatively full box, the pallets can be positioned in a situation in which a lot of time-lag arises. To test the performance of the model, the situations will be divided into three different categories[21].

First some cases will be presented where 5 'normal' case situations will be calculated. This collection of cases will be a summation of real cases in which just a small amount of time-lag had influence on the timespan of the total outfeed process. This amount of time-lag will be between 10% and 20% of the total timespan. The cases will present the performance of the model in situations that occur frequently in the warehouse.

The second group of cases represent outfeed situations that only appear a limited amount of times in the warehouse. For example 1 out of 8 outfeed trucks has the amount of time-lag similar to this group of cases. However, this small amount of outfeed trucks contributes a big part of the total time-lag. The amount of time-lag in these situations is between 20% and 30% of the total timespan.

The last group of cases are situations that have a low chance of actually occurring in the warehouse. However, these are good examples to determine what the difference in outfeed times can be between the old system and the newly developed model. These situations will probably provide the largest difference between the current situation and the future situation when implementing the new model with a total time-lag of 30^{+} %

[21] http://www.umich.edu/~elements/05chap/html/05prof2.htm

The first case of the normal scenarios will be elaborated on extensively to explain the way these scenarios are tested. The outcomes will be presented in the table simulation results for every case. The emulation tool of ORTEC is used to acquire the needed test results. The cases are first run in the old model to determine the total timespan needed in the current warehouse. This will be presented in the table as the 'Emulation Old' times. The next step is to put these actions into the model, which will result in the MIP Model (optimization) time. This is the minimal timespan found by the new model. The output of the model will be a new pallet sequence, which can then be used as the input for the emulation tool. The outcome of this emulation tool is the timespan needed if the new model is implemented and will be presented as the 'Emulation MIP' times. These Emulation MIP times can be compared with the Emulation Old times in order to determine the difference between the situation with and without the implementation of the new model. Finally the calculation time of the model is presented, providing the time needed by the model to come to a solution. The machine configuration on which these tests are performed is given in appendix D.

4.2 'Normal' (most common) case scenarios

This paragraph will present a total of 5 cases with the simulation outcomes of these situations. The cases discussed are called normal cases, which means these or similar cases often occur in the warehouse. The cases are called DB1 till DB5_Normal, where DB stands for Database. The first case will be elaborated on in this paragraph, for the other cases just the outcomes will be shown and compared (the information on the start position will be shown in appendix E).

<u>**Case: 1**</u>

The first figure presents the 4 levels of the warehouse, with the pallets selected for outfeed (orange, dark border) and the pallets selected for digging-out (light-orange, red number). These 'dig-out pallets' will in this case be moved to the neighboring lane. The number inside each pallet represents the action number at which the pallet is moved in the 'old' situation.

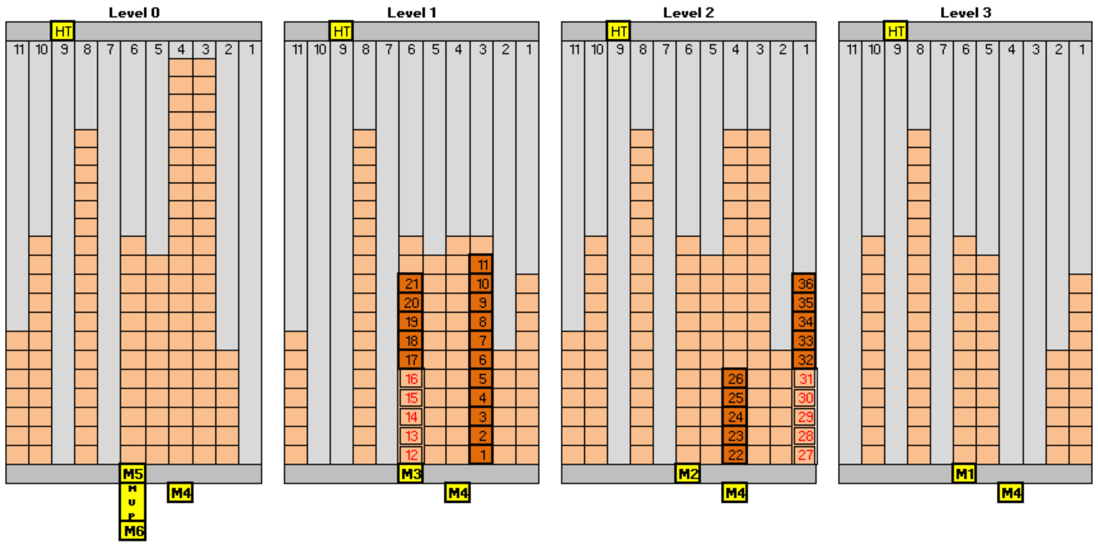


Figure 4.1: Start situation case: 1

The actions for this outfeed situation are given in table 4-1. These are the actions generated by WPMS without using the model. The actions correspond with the numbers inside the pallets in figure 4.1.

Table 4-1: WPMS sequence

ActionID	From	To
1	L1-03	DropPoint
2	L1-03	DropPoint
3	L1-03	DropPoint
4	L1-03	DropPoint
5	L1-03	DropPoint
6	L1-03	DropPoint
7	L1-03	DropPoint
8	L1-03	DropPoint
9	L1-03	DropPoint
10	L1-03	DropPoint
11	L1-03	DropPoint
12	L1-06	L1-05
13	L1-06	L1-05
14	L1-06	L1-05
15	L1-06	L1-05
16	L1-06	L1-05
17	L1-06	DropPoint
18	L1-06	DropPoint
19	L1-06	DropPoint
20	L1-06	DropPoint
21	L1-06	DropPoint
22	L2-04	DropPoint
23	L2-04	DropPoint
24	L2-04	DropPoint
25	L2-04	DropPoint
26	L2-04	DropPoint
27	L2-01	L2-02
28	L2-01	L2-02
29	L2-01	L2-02
30	L2-01	L2-02
31	L2-01	L2-02
32	L2-01	DropPoint
33	L2-01	DropPoint
34	L2-01	DropPoint

Table 4-2: Scheduler sequence

ActionID	From	To
1	L2-04	DropPoint
2	L2-04	DropPoint
3	L2-04	DropPoint
4	L1-03	DropPoint
5	L1-03	DropPoint
6	L1-06	L1-05
7	L2-04	DropPoint
8	L2-04	DropPoint
9	L1-03	DropPoint
10	L1-06	L1-05
11	L2-01	L2-02
12	L1-06	L1-05
13	L1-03	DropPoint
14	L1-03	DropPoint
15	L1-06	L1-05
16	L1-03	DropPoint
17	L1-03	DropPoint
18	L1-03	DropPoint
19	L1-06	L1-05
20	L1-06	DropPoint
21	L1-06	DropPoint
22	L1-03	DropPoint
23	L2-01	L2-02
24	L1-06	DropPoint
25	L1-06	DropPoint
26	L2-01	L2-02
27	L1-03	DropPoint
28	L1-03	DropPoint
29	L2-01	L2-02
30	L2-01	DropPoint
31	L1-06	DropPoint
32	L2-01	DropPoint
33	L2-01	DropPoint
34	L2-01	DropPoint

These actions are then used as input for the emulation tool of ORTEC, to come to a total timespan of 1402 sec. The minimum amount of time needed for the processing of this situation can easily be calculated: every pallet needs 43 second on the MUP + the time it takes to get the first pallet on the MUP, this leads to a minimal timespan of 1168 seconds. This is much less than the 1402 seconds in the previous situation. It can be concluded that the amount of time-lag in the current system is 1402 – 1168 = 234 seconds for this specific case.

These same actions will be put into the model which leads to a new sequence for the actions. The new sequence is given in table 4-2. When the actions are executed in this new sequence, the expected timespan is given. In this case the calculated timespan is 1168 seconds. This can also be seen in figure 4.2 which displays all the actions in a Ganttchart.

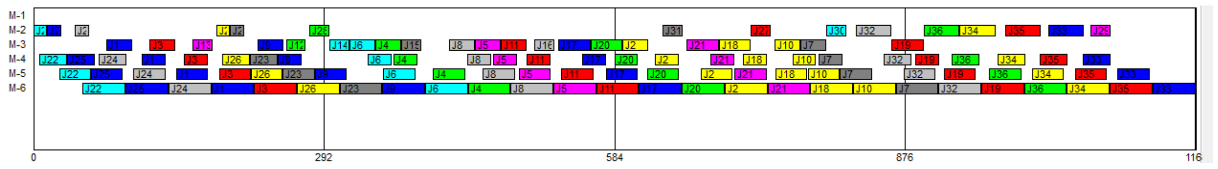


Figure 4.2: Ganttchart case 1

The new sequence is then again used as input for the emulation software. The model gives a minimal timespan of 1168 seconds. When this case is ran in the emulation software the total timespan is 1168 as well. This similarity of timespans can be seen as a first indication that the resource times used in the emulation software are equal to the resource times used in the model. If these values would have a small deviation, this can be explained by the fact that the calculation times in the model are rounded numbers instead of the numbers in the emulation software, which are accurate at the millisecond. In order to compare the old and the new situation the focus will be on the emulator results, because the emulation software will make use of the exact same resource times in both the old and the new situation. Therefore the 'Emulation Old' and the 'Emulation MIP' have the exact same external conditions. The test results of the first situation are given in the table.

Table 4-3: Simulation results case 1

Simulation Results [seconds]		
MIP Model	=	1168
MIP Calculation Time	=	40
Emulation Old	=	**1402**
Emulation MIP	=	**1168**

These results show the difference of the total timespan of 236 seconds. This difference in outfeed times can be explained by the large amount of time-lag that was present in this situation. The amount of time-lag in the new situation can easily be determined by comparing the calculated minimal amount of timespan with the actual timespan needed. In this specific case the minimal timespan is equal to the timespan determined by the model. This means the model has found an optimal sequence in which no more time-lag is present. This optimality was the predicted outcome, because the model only presents a solution if it has found an optimal solution.

In figure 4.3 an example is shown of the convergence of the upper- and lower bound of the CPLEX 12.5 solver. This figure shows the progress of the solver, with the calculation time on the horizontal axis and the objective value on the vertical axis. The solver starts with an infinite upper bound and a lower bound of 0. From these starting values the first calculated values of the upper and lower bound can be seen in the figure. This upper bound is the first solution found by the model. The lower bound represents the total time necessary to process all the actions on the machine with the most processtime in this situation. In this case this is the MUP (M-6). The solver will calculate on both the upper bound and the lower bound to determine the point of intersection. This point of intersection represents the solution found by the model. The main conclusion from this graph is the fact that the upper bound has found the minimal timespan way before the lower bound does.

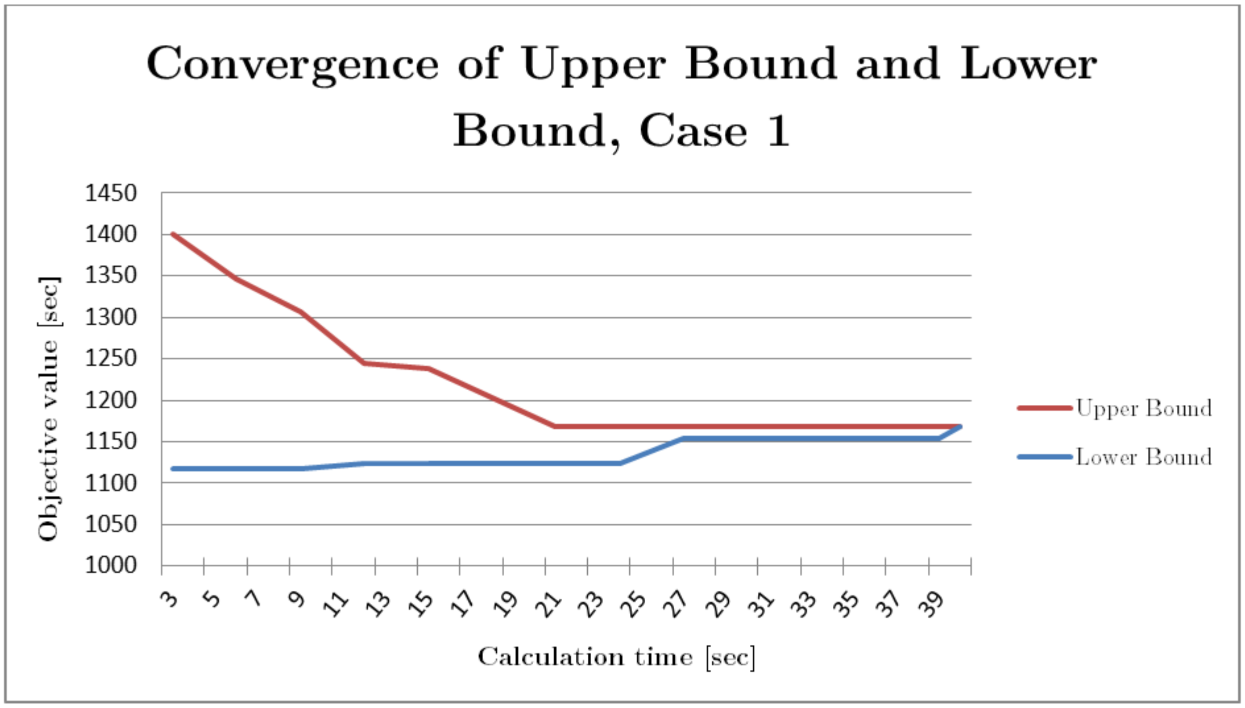


Figure 4.3: Convergence of Upper- and Lower bound

Case: 2

The next cases will be done in the same way as the first case has been done; only they will be elaborated on a bit less extensively. The start situations of these simulations can be seen in appendix E and only the model and simulation results will be given.

Case 2 has a minimal calculated timespan of 1167. Simulating the pallet sequence in the emulation tool generates timespan of 1400 seconds. With the use of the model, the new sequence needs a timespan of 1167 seconds, equal to the calculated timespan.

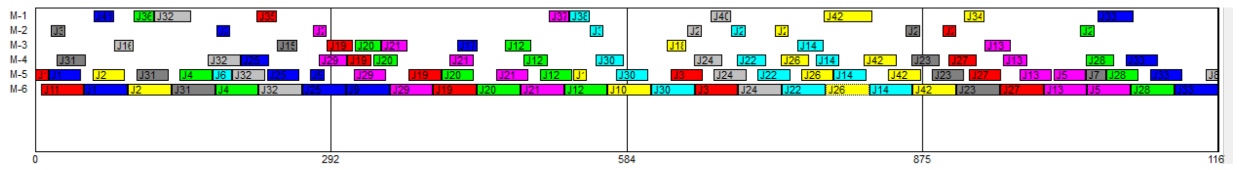


Figure 4.4: Ganttchart case 2

Finally the sequence is used as input in the emulation tool, which generates a timespan of again 1167 seconds. The result can be seen in table 4-4.

Table 4-4: Simulation results case 2

Simulation Results [seconds]		
MIP Model	=	1167
MIP Calculation Time	=	45
Emulation Old	=	**1400**
Emulation MIP	=	**1167**

The difference between the non-optimized and the optimized situation is: 1400 - 1167 = 233 sec.

Case: 3

The minimum timespan is again calculated by the total amount of time the MUP is busy, plus the time it takes for the first pallet to arrive at the MUP. In this case the minimum timespan is 1443 sec. When running the emulation software the old situation has a total timespan of 1672 sec. A new sequence is determined by the model, which can be seen in figure 4.5.

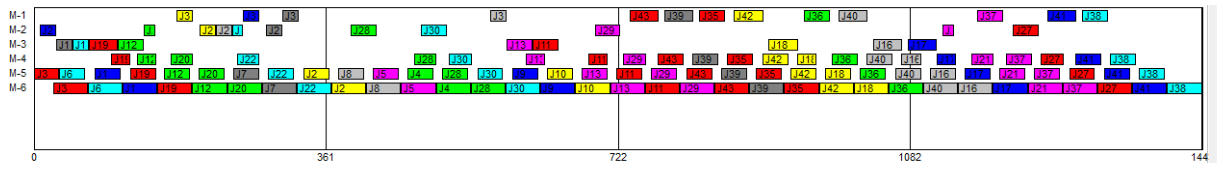


Figure 4.5: Ganttchart case 3

The model calculates a total timespan of 1443 seconds, which is equal to the calculated minimum timespan. This means the model optimized the situation. The results are shown in the table.

Table 4-5: Simulation results case 3

Simulation Results [seconds]		
MIP Model	=	1443
MIP Calculation Time	=	43
Emulation Old	**=**	**1672**
Emulation MIP	**=**	**1444**

The results show a difference of 1672 - 1444 = 228 sec. This extra second needed in the emulation MIP, can be explained by the small rounding errors (see paragraph 3.7).

Case: 4

In this case the minimal timespan is 1198. The timespan needed by the emulation software is a total of 1340 sec. In this case the total amount of time-lag can be calculated: 1340 – 1198 = 142 sec. When running this case with the model, a minimal timespan of 1202 can be achieved. This means there is still 4 seconds of time-lag in the system. This can be explained by the fact that it is not possible to dig-out the pallets on L0 within the 43 seconds the MUP takes to move a pallet to the outfeed point. Therefore in this situation the optimum found by the model is not equal to the calculated lower bound. The lower bound calculation is wrong in this case. This difference between the calculated lower bound and the actual lower bound is from now on called the technical gap. The actual lower bound in this case will be the calculated lower bound + the technical gap. The sequence can therefore be seen as optimized if the solution determined by the model is equal to the actual lower bound. The starting position of the pallets can be seen in appendix E.

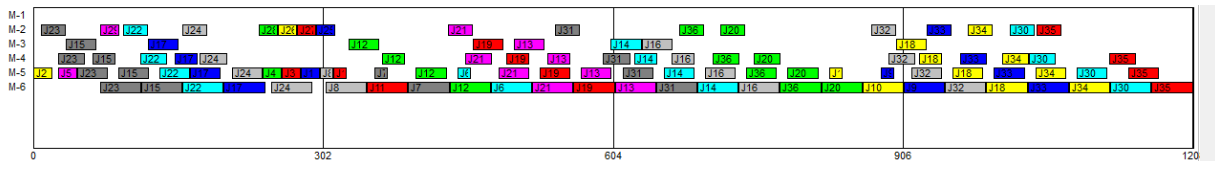


Figure 4.6: Ganttchart case 4

The result of the simulation can be seen in table 4-6.

Table 4-6: Simulation results case 4

Simulation Results [seconds]		
MIP Model	=	1202
MIP Calculation Time	=	511
Emulation Old	**=**	**1340**
Emulation MIP	**=**	**1202**

The results show a difference of 1340 - 1202 = 138sec.

<u>Case: 5</u>

The minimal timespan is calculated on 1142 seconds. When running the case with the standard sequence, a total timespan of 1280 is needed. The developed model generates a sequence which has a total timespan of 1142 seconds; no more time-lag is present inside the system. This can also be seen in figure 4.7.

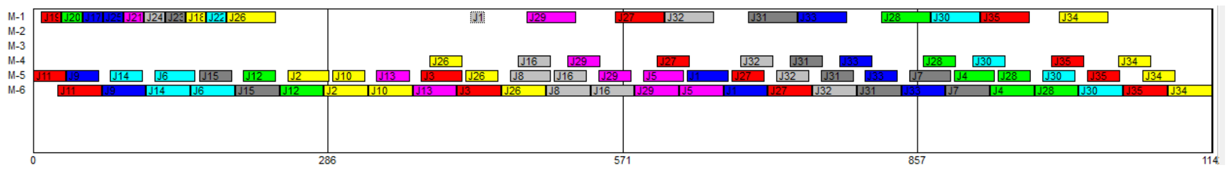


Figure 4.7: Ganttchart case 5

The simulation results are given in table 4-7.

Table 4-7: Simulation results case 5

Simulation Results [seconds]		
MIP Model	=	1142
MIP Calculation Time	=	8
Emulation Old	=	**1280**
Emulation MIP	=	**1142**

The results show a difference of 1280 - 1142 = 138 sec.

4.3 Bad (less common) case scenarios

The cases described in this paragraph are less common situations. The frequency at which these cases actually occur in the warehouse is less than the ‘Normal’ cases described in the paragraph above. The frequency at which similar cases arise in the actual warehouse is about once or twice a day in an average situation.

<u>Case: 6</u>

This case has a minimal calculated timespan of 1329 seconds. When simulating it in the normal sequence, a timespan of 1812 seconds is needed which means there is lots of time-lag present in the system. Simulated in the new sequence, this case takes 1329 seconds as well; again the situation is optimized.

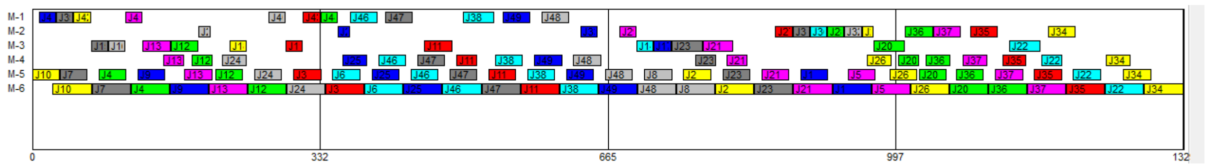


Figure 4.8: Ganttchart case 6

Running the new sequence in the emulation tool, leads to a timespan of 1329 seconds.

Table 4-8: Simulation results case 6

Simulation Results [seconds]		
MIP Model	=	1329
MIP Calculation Time	=	203
Emulation Old	=	**1812**
Emulation MIP	=	**1329**

The results show a difference of 1812 - 1329 = 483 sec.

Case: 7

This case has a minimal calculated timespan of: 1154 sec. When the actions are used as input into the emulation tool, a total amount of 1476 sec is needed to execute all the actions. Again this action sequence is recalculated in the model, the outcome is given in figure 4.9.

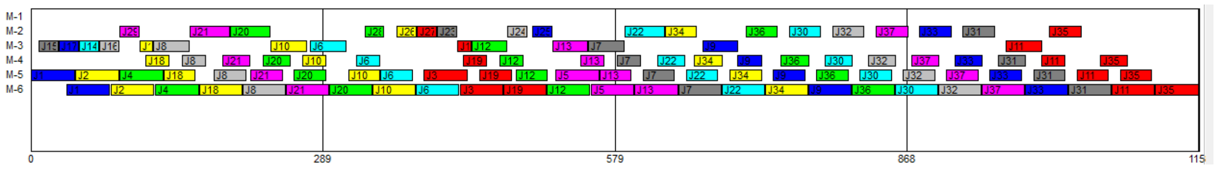


Figure 4.9: Ganttchart case 7

The minimum timespan needed according to the model is 1158. Therefore in this case the model is not able to get to the calculated minimum. Again a technical gap is present in the process of 4 seconds. In this case the reasons for this technical gap are the 5 pallets on level-0 that have a resource time of 44 seconds on M-5. This 44 seconds is one seconds longer than the MUP is busy with processing a pallet. This will lead to the total technical gap of 4 seconds (not 5 seconds, because the first pallet can be a pallet on level 0). When using the new sequence as input for the emulation model, the model has a simulated total timespan of 1158 seconds. The test results are given in the table.

Table 4-9: Simulation results case 7

Simulation Results [seconds]		
MIP Model	=	1158
MIP Calculation Time	=	104
Emulation Old	**=**	**1476**
Emulation MIP	**=**	**1158**

The results show a difference of 1476 - 1158 = 318 sec.

Case: 8

This case has a calculated timespan of1124 seconds. The normal sequence generates a timespan of 1368 seconds. This means lots of time-lag is present inside the system. The model generates a timespan of 1124 seconds in this situation, the situation is optimized. Figure 4.10 shows the output of the model.

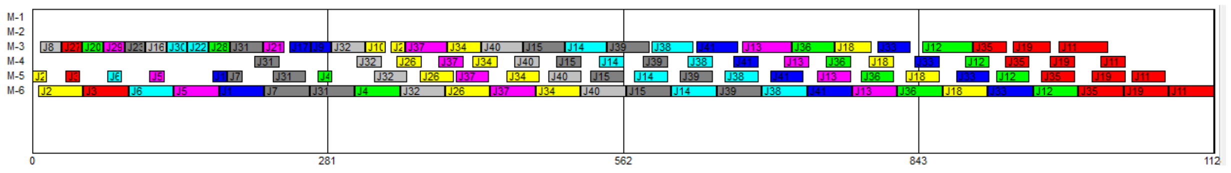


Figure 4.10: Ganttchart case 8

Using this outfeed as the infeed for the emulation tool, the timespan of 1124 is determined. This can be seen in table 4-10.

Table 4-10: Simulation results case 8

Simulation Results [seconds]		
MIP Model	=	1124
MIP Calculation Time	=	44
Emulation Old	**=**	**1368**
Emulation MIP	**=**	**1124**

The total difference between the non-optimized and the optimized situation is: 1368 - 1124 = 244 sec.

4.4 Worst case scenarios

In this paragraph some cases will be presented that do not represent any real scenarios. These cases are specifically made-up to determine how well the model works in scheduling the sequence of the actions. These cases could appear in the warehouse, however the possibility is small. All the cases are shortly discussed and again the starting situation of all these cases can be found in the appendix E.

Case: 9

Case 9 represents a situation in which on level 0, 8 pallets are at the front of lane 8. On level 3 there is one pallet at the front of lane 6 and one pallet in lane 4. This pallet in lane 4 is behind 13 pallets that need to be dug-out (see appendix E). When running this case, the normal sequence generates a total timespan of 708 seconds, while the calculated optimum is 454 seconds. The model has optimized this situation to a timespan of 454 seconds.

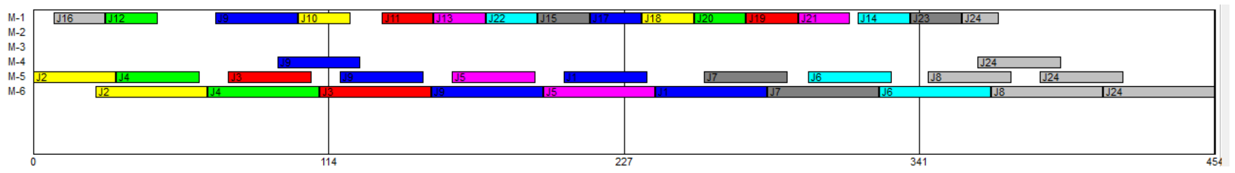


Figure 4.11: Ganttchart case 9

Table 4-11 presents the results.

Table 4-11: Simulation results case 9

Simulation Results [seconds]		
MIP Model	=	454
MIP Calculation Time	=	25
Emulation Old	=	**708**
Emulation MIP	=	**454**

The total difference between the situation with and without the scheduler is: 708 – 454 = 254 sec. This is a timespan reduction of almost 36%.

Case: 10

This case in an example of a situation where 15 pallets are at the front of the lane on level 0, while there are 16 dig-out actions on level 3 on two separate lanes. Simulating the situation leads to a timespan of 1764 seconds. Running this case in the model, gives a minimized timespan of 1150 seconds. This is similar to the calculated minimal timespan.

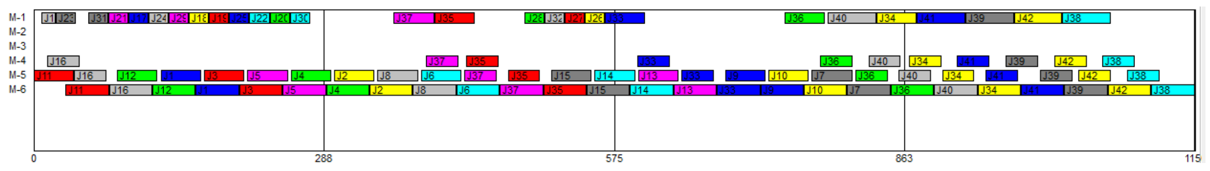


Figure 4.12: Ganttchart case 10

Running the new sequence in the emulation tool, leads to a timespan of 1150 seconds.

Table 4-12: Simulation results case 11

Simulation Results [seconds]		
MIP Model	=	1150
MIP Calculation Time	=	45
Emulation Old	=	**1764**
Emulation MIP	=	**1150**

The results show a difference of 1764- 1150= 614 sec and a total of almost 35%.

Case: 11

The final case is a similar case as the previous one and is used to show how big the difference can be between the original sequence and the sequence developed by the model. This case has dig-out actions on level 1 on two separate lanes. The simulation provides a timespan of 1300 seconds. Running the situation with the model reduces the timespan to 892 seconds as can be seen in the figure. The calculated minimal timespan is 892 as well.

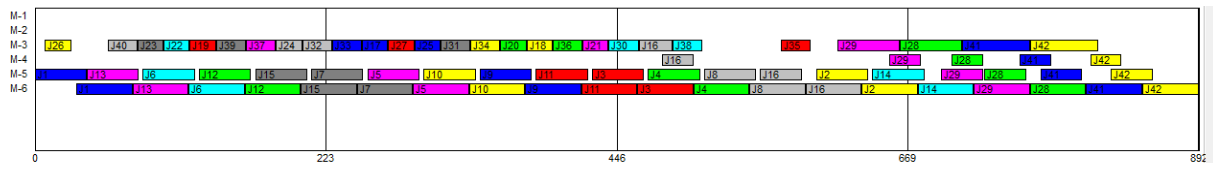


Figure 4.13: Ganttchart case 11

The results can be seen in table 4-13.

Table 4-13: Simulation results case 11

Simulation Results [seconds]		
MIP Model	=	892
MIP Calculation Time	=	40
Emulation Old	=	**1300**
Emulation MIP	=	**892**

The optimized sequence put into the emulation tool provides a timespan of 892 seconds. This is a reduction of almost 32% of the time-span and a 100% reduction in time-lag.

4.5 (Intermediate) Result summary

This paragraphs sums up the results found in the cases described in the previous paragraphs into table 4-14. The first column contains the case number, the second column gives the timespan reduction in the optimized situation. In the third column the total amount of timespan reduction is given in seconds. The fourth column describes the technical gap which is still present inside the system when the outfeed actions are optimized. This is the difference between the calculated minimum and the optimization of the model. An explanation for this technical gap is the impossibility of the pallets to be at the MUP within the 43 seconds. This can either be because pallets need to be dug out on level 0, or there are pallets on level 0 with a longer travel time than 43 seconds. This time can be seen as unavoidable time-lag in the system. The fifth column gives the deviation from the technical gap. This value represents the deviation of the outcome of the model in comparison with the possible minimum timespan, the amount of time which the situation can still be improved. In the cases 4 and 7, a technical gap of 4 seconds is present with a deviation from this technical gap of 0. Therefore the situation can be seen as optimized while the calculated minimum is not achieved. The last column in the table represents the calculation time of the model. These calculation times vary a lot, from 8 seconds up to 511 seconds dependent on the case.

Table 4-14: Results

Results

Case	Timespan reduction Optimized [%]	Timespan reduction [sec]	Technical gap [sec]	Deviation from technical gap [sec]	Calculation time [sec]
1_Normal	16.8	236	0	0	40
2_Normal	16.6	233	0	0	45
3_Normal	13.7	228	0	0	43
4_Normal	10.1	136	4	0	511
5_Normal	10.8	138	0	0	8
6_Bad	26.7	483	0	0	203
7_Bad	21.6	318	4	0	104
8_Bad	20.8	244	0	0	44
9_worst	35.9	254	0	0	25
10_worst	34.8	614	0	0	45
11_worst	31.4	408	0	0	40

With these cases an estimation can be given on the time this new model will actually reduce the outfeed times of the Optilogx box. However, this time is very dependent on the situation inside the box (storage capacity utilization) and the placement of the selected pallets.

Therefore the frequency in which similar situations occur as described in the cases, cannot be exactly determined. The number of outfeed orders per time equivalent also has a high deviation, dependent on the season of the year, the week of the month and even on the day of the week. However, with the experience of the Hellendoorn Optilogx box and in consultation with the OWCO team a rough estimation can be made to determine the order of magnitude of the time-reduction at the Hellendoorn warehouse:

In about 50 % of the outfeed orders at the real warehouse, a situation as represented in the first five 'normal' cases occurs. For these cases an average timespan reduction of 15% is taken. The cases presented as the 'bad' situations, occur in the warehouse for about 10% of the time and have an average timespan reduction of about 25%. The 'worst' cases only occur 2 % of the time, but have an average timespan reduction of about 35%. The remaining 38% of the time, situations occur in which a smaller amount of time-lag is present than in the discussed cases. For these cases an average timespan reduction of 5% is taken. A simple calculation leads to the total timespan reduction of 12.6% on an average outfeed order. The percentages are presented in table 4-15.

Table 4-15: Time-lag overview

Time-lag in the different situations

Situation	**Average presence** [%]	**Time-lag** [%]	**Average reduction** [%]
Normal	50	15	7.5
Bad	10	25	2.5
Worst	2	35	0.7
Rest	38	5	1.9
Total	100	80	12.6

However there is a problem; the calculation times used in these cases are unacceptable for the current warehouse. An orderbatch will be requested at the moment the truck is ready at the loading dock, and therefore there is no time available for the calculation of the optimal sequence. In order to make the scheduler work in a real warehouse, a maximum calculation time can be set in the model. In consultation with ORTEC a maximum calculation time of 10 seconds is determined. These 10 seconds can be used by the scheduler to determine the best solution found in this calculation time. When an optimized solution is found within the 10 seconds, the model will just give the result at that time. In the next paragraph the same cases will be run in order to determine the results, when taking the maximum calculation time restriction of 10 seconds into account.

4.6 Model calculations with time restriction

As can be seen in the previous paragraph, the calculation time of the model differs a lot. In the real-time situation it is not preferred to have the model taking several minutes to calculate the right order of outfeed pallets. This because the orderbatch (containing between the 26 and 33 pallets) will be requested at the time the truck is ready at the loading dock. In this case the assumption is made that the orderbatch request time cannot be earlier. Therefore a maximal calculation time restriction has to be put into the model to make it feasible in the real warehouse. This calculation time is determined in the cooperation with ORTEC and is defined as a maximum time of 10 seconds. Again the previously presented cases will be run in order to show the difference with the fully optimized situations. The separate test results are presented in appendix F. In this paragraph only the summation of the test results and the comparison with the test result without the maximum calculation time restriction (seen in the previous paragraph) will be presented.

Table 4-16: Results with time restriction

Results time restriction

Case	% of previous timespan reduction	% of timespan reduction with restriction	Timespan reduction [sec]	Deviation from technical gap [sec]	Calculation time [sec]
1_Normal	16.8	14.7	207	29	10
2_Normal	16.6	16.6	232	1	10
3_Normal	13.7	13.7	228	0	10
4_Normal	10.1	9.0	120	16	10
5_Normal	10.8	10.8	138	0	8
6_Bad	26.7	20.8	376	107	10
7_Bad	21.6	21.3	314	4	10
8_Bad	20.8	20.4	238	6	10
9_worst	35.9	33.9	240	14	10
10_worst	34.8	34.8	614	0	10
11_worst	31.4	31.4	408	0	10

The table above sums up the test results of the test cases when the new model can fully optimize the sequence (% of previous timespan reduction), and the results determined with the calculation restriction of 10 seconds (% of timespan reduction with restriction). The difference between these values can be seen as the percentage of time-lag still present inside the system when the calculation time restriction is implemented. These values show a difference of only a small percentage (from 0% to a peak of 6.1%), which means most of the time-lag is still reduced in this new situation with the calculation time restriction. In about a third of the cases the values are actually the same as the previous unlimited calculation time. This means the solver had already found the optimal solution, but was still calculating the lower bound to climb up towards the upper bound (see also the explanation in case 1). The fourth column presents the timespan reduction in seconds, with the same small deviation from the previous test result. In the fifth column the deviation from the technical gap (as explained in the previous paragraph) is given in seconds. This is the actual time-lag in seconds that is still present in the warehouse situations. As can be seen in the table, these values range from 0 to a maximum of 107 seconds, with an average of only 16 seconds of time-lag still present in the system. Compared to the situation before, which had an average time-lag of 300 seconds, this is still a huge improvement. The final column is the calculation time of the model, which has a maximum value of 10 seconds in this situation.

The outcomes of these tests can be seen as very positive. In the situation of no calculation time restriction, the model optimizes the sequence. This means if the orderbatches can be announced earlier than at the last moment, for example when the truck drives onto the warehouse site, the outfeed process can be fully optimized. Another possibility for the system to create some calculation time, is to start the outfeed procedure with the pallet that is closest to the MUP at the moment that an orderbatch is put into the system. In the time the warehouse is busy with moving the pallet to the outfeed, the scheduler can calculate an optimal solution for the rest of the pallets in the orderbatch. More testing is required to determine whether or not this will perform better than to just wait 10 seconds at the beginning of the outfeed procedure. On the other hand when taking the calculation restriction of 10 seconds into account, the model still performs very well in reducing the time-lag at the outfeed process. These test results can lead to the recommendation towards ORTEC which implies on the high amount of time-lag reduction that can be achieve by implementing this scheduler at the Hellendoorn warehouse.

5 General Conclusion

In the previous chapter several cases have been discussed. First some cases in which the solver was not time restricted in any way and secondly the same cases in which the solver had a total time restriction of 10 seconds. This final chapter will present the conclusions drawn from this research, along with the recommendation towards ORTEC. Finally a reflection will be presented in which the positive and negative experiences in this research will be discussed.

5.1 Conclusion

In this report a solution for the problem at the fully automated warehouse in Hellendoorn is sought. This problem is the amount of time it takes for the Optilogx box to transport a selected orderbatch to the outfeed of the system, from where the pallets will be loaded into a truck. This outfeed-time is in some cases (a lot) more time consuming than the technical capacity of the system prescribes: The technical outfeed capacity of the Optilogx box is not reached. The analysis showed that the main reason for this problem is the fact that the 'optimizer' of the OWCO software does not take resources or time into account. The main research question formed for this project is:

How can the technical capacity of an automated warehouse be reached by minimizing the time-lag caused by the Optilogx box, when focusing on the optimizer of the OWCO system?

The problem inside the Optilogx box can be described as a combination between a standard permutation flowshop problem and a jobshop problem without pre-emption. This problem is NP-complete for m = 3 machines or more. Therefore no standard available algorithm can be used to optimize this problem in the polynomial time. A model is developed to reduce the amount of time-lag inside the system in the outfeed procedure of the automated warehouse. This outfeed time reduction is achieved by rescheduling the sequence of the outfeed actions. The objective function of the developed model is to minimize the variable of the total timespan. This variable is minimized while taking the values of the parameters and sets as a starting point and satisfying the conditions of the constraints. The developed model will make use of the CPLEX 12.5 solver in order to solve the MIP problem described by the outfeed procedure of the warehouse.

The developed model is tested on two configurations, each containing the same test scenarios with the amount of timespan reduction and the calculation times as the key performance indicators. At the Hellendoorn warehouse the orderbatch is requested at the moment a truck is ready at the outfeed. Therefore, the calculation times of the model cannot be neglected. The first test configuration is a test in which the assumption is made of an unlimited amount of calculation time. The second test configuration has a calculation time with a maximum of 10 seconds to determine the outfeed sequence. In this case the model will not present the optimized solution, but the best possible solution found within these 10 seconds.

The results of both test scenarios are promising. In the first scenario the outfeed process is fully optimized, meaning no more time-lag is present in these outfeed procedures. The big disadvantage is the amount of calculation time needed in some situations, varying between 8 and 511 seconds. The second scenario will not fully optimize the outfeed sequence in every case, but will still reduce the amount of time-lag from an average test result of 21.7% time-lag to an average of only 1.1% time-lag. In the real warehouse this will lead to an estimated timespan reduction between the 10% and 15% dependent on the capacity utilization on an average day at the warehouse in Hellendoorn. In the current situation the outfeed times are the bottleneck in the entire outfeed procedure. Therefore the reduction in outfeed times will have the same amount of direct reduction on the loading times of the trucks. On an average day at the Hellendoorn warehouse, delay of outfeed trucks as a result of long outfeed times is no exception. Neither are the additional expenses these delays result in. Therefore the direct 10% to 15% time reduction will make a lot of difference in the situation of the Hellendoorn warehouse.

5.2 Recommendation

These test results can lead to the recommendation towards ORTEC which implies on the high amount of time-lag reduction that can be achieved by implementing the developed scheduler at the Hellendoorn warehouse. The recommended place for implementation of the scheduler will be inside the top layer of software (WPMS) in the OWCO software.

This scheduler is specifically configured to work at the outfeed procedure of the Optilogx box as can be found at the Hellendoorn warehouse. The generic design however will make it possible for the model to also work for an Optilogx box with a different configuration (a different amount of lanes, shuttles or VTs). This can be achieved with the necessity of only minor adjustments. The model could also be implemented for the outfeed procedure of an Optilogx-satellite warehouse, again with only minor adjustments in the configuration of the model. The remark hereby is that the model only works with scheduling of the outfeed processes.

Some more research has to be done on the amount of calculation time needed by the scheduler, in order to come to a (close to) minimal solution. The tradeoff will have to be made between a long total timespan of outfeed actions and a short calculation time, or a shorter total timespan, but a longer calculation time. The best option is different for each situation and is dependent on the start situation inside the Optilogx box, as well as the amount of pallets selected for the orderbatch. Another possibility is to have the orderbatch requests available prior to the moment the pallets are needed at the outfeed. This will create some 'extra' time in which the scheduler can determine the optimal sequence. A possible solution would be to create the orderbatch request at the moment the truck drives onto the warehouse site. It could also be possible to start the outfeed situation with the pallet closest to the MUP at the moment an orderbatch is put into the system. In the time this first pallet is moved to the MUP, the scheduler can calculate the optimal sequence for the rest of the pallets. Some more testing is required in order to determine whether or not this will outperform the situation of a 10 second waiting time.

For the tests described in chapter 4, the standard configuration of the CPLEX 12.5 solver is used. Some research is recommended to determine whether or not it is possible to decrease the calculation time by tweaking the solver configurations. Another possibility to decrease the calculation time is to do a lower bound calculation and let the solver start at this calculated value instead of starting from a lower bound of 0.

An even better situation is when the Optilogx box is perfectly sorted every time an orderbatch is requested. In such a situation no time-lag will be present inside the system and a scheduler is therefore not required. While the situation of a perfectly sorted box is almost impossible to maintain for 24 hours a day, the recommendation is to keep working on the functionality that broods over this sorting: housekeeping. I would also like to recommend ORTEC to look at the possibility of replacing the outfeed MUP at the Hellendoorn warehouse by a simple chain conveyer lane. This would not only increase the outfeed capacity of the warehouse by a lot, but will also have a positive effect on the reliability of the outfeed procedures.

5.3 Reflection

Looking back on the entire research, I can conclude that it went pretty well. The path that was going to be followed and the results desired by ORTEC were already clear in the first stages of the project. There were no problems at the beginning of the research between the desired results ORTEC wanted in contrast to the desired results of the TU Delft. Beforehand I thought it would only be a matter of time before small disagreements arose between the two parties. The opposite proved to be true, and also later on in this research the collaboration between ORTEC and the TU Delft worked well. The process towards the development of the scheduler in which lots of knowledge in the field of automated warehouses was collected went prosperous.

The methods chosen to analyze the current situation at both the warehousing market and the specific Optilogx box worked well and provided a lot of useful information. This analysis phase resulted in the research and sub-research questions needed in order to make a start with the development of the model. The program used for the development of the model (AIMMS) was new to me, and it took some time to get used to the program. I'm sure the model could be developed a bit faster in AIMMS, if it was developed by someone with more AIMMS experience. However, the process of developing the model did proved to be very useful to me, especially when looking back at the learning curve involved.

The next step was to validate en verify the model, which proved to be a bit of a problem. I did not have a lot of experience with model validation or verification, which led to some problems in this process. Thanks to extensive discussion with my supervisors this chapter was improved. Again a lot was learned from this process, and I am confident to perform more model validations and verifications in the future.

Finally the performance of the model needed to be tested. From the start it was clear that the emulation tool of ORTEC could be used for these performance tests. This provided a solid basis for the development of the test scenarios. The testing itself went well, the only negative aspect on the whole testing procedure was the amount of time it took to perform all the tests. This took longer than expected, which led to some time scheduling problems in the second half of the research.

The development of the report proved to be a process with ups and downs. The first half of the report went pretty smoothly, but in the second half some parts had to be rewritten. Especially in the final weeks before the hand-in date some parts of the report had to be revised. A main reason for this was the useful input of Mark Duinkerken on improvements on some parts of the report. On the other hand it became more clear that there is a big difference in the views on the report between the different faculties. Looking back at this second half of the research, I would do it a bit differently. I would have liked to have included Mark Duinkerken far earlier in the process. Especially the discussion about the mathematical formulation and the verification and validation was very useful. I would have also started a lot earlier with the development of the different testing scenarios and the actual testing of the model.

If I had some more time for this research I would have really liked to look into the situation of the Hellendoorn warehouse without the MUP at the outfeed. Replacing the MUP for a faster outfeed resource would have been a very different situation with a different bottleneck for the outfeed procedures.

I have enjoyed doing this research a lot. I've learned a whole lot, also on the subject of working in a business environment. Another great experience of the assignment was meeting new people and getting familiar with ORTEC, in particular the OWCO department. The experience of this assignment is the start of my career, a new chapter of my personal development which will continue for the coming years.

References

Websites

[1] http://www.spie-nl.com (accessed December 2012)

[2] http://www.spie-nl.com/en/divisions/spie-industry/products-and-services/optilogx.html (accessed December 2012)

[3] http://eyemerge.nl/verschillende-situaties (accessed December 2012)

[4] http://www.directindustry.com/prod/jungheinrich-1078.html (accessed December 2012)

[5] http://www.bruynzeel-storage.nl/Producten/Verrijdbare-archiefkasten/Compactus-Manual/ (accessed December 2012)

[6] http://www.daifukuchina.com (accessed January 2013)

[7] http://ttmnl.wpengine.netdna-cdn.com/wp-content/uploads/2007/01/07ttm12_disrep.pdf (accessed December 2012)

[8] http://www.psb-gmbh.de/ (accessed January 2013)

[9] http://www.rotra.nl/bicycle/ (accessed January 2013)

[10] http://automatedparkingsystems.blogspot.nl/2013/06/automated-car-parking-history.html (accessed February 2013)

[11] http://www.computable.nl/artikel/nieuws/overheid/2255264/1277202/garage-plet-auto.html (accessed January 2013)

[12] http://continuingeducation.construction.com/article_print.php?L=316&C=942 (accessed February 2013)

[13] http://www.thecoolist.com/autostadt-automated- parking-garage-towers-2/ (accessed February 2013)

[14] http://www.parkmatic.com/#!___automated/optima-parking (accessed January 2013)

[15] http://www.wics.nl/ (accessed December 2012)

[16] http://www.storaxsupplies.com/ (accessed December 2012) (accessed January 2013)

[17] http://www.123rf.com (accessed February 2013)

[18] http://www.businessdictionary.com/ (accessed March 2013)

[19] http://plato.asu.edu/ftp/feas_bench.html (accessed June 2013)

[20] http://plato.asu.edu/ftp/feas_bench_logs/ (accessed June 2013)

[21] http://www.umich.edu/~elements/05chap/html/05prof2.htm (accessed August 2013)

[22] http://www.ortec.nl (accessed December 2012)

[23] http://www.heezik.nl (accessed December 2012)

Literature

[1] Alharkan, Ibrahim M (1999). Algorithms for Sequencing and Scheduling, University of Oklahoma.

[2] Brain Morriss S. (1999) Programmable Logic Controllers, Prentice Hall PTR Upper Saddle River, NJ, USA, ISBN:0130955655

[3] Burkard, R. E. Linear Assignment Problems and Extensions, 1–54.

[4] Constantino, A. (2011). A heuristic algorithm for nurse scheduling with balanced preference satisfaction. *… in Scheduling (SCIS …*, 39–45. Retrieved from

[5] Cotta, C., & Fern, A. J. (n.d.). Memetic Algorithms in Planning , Scheduling , and Timetabling.

[6] Dantzig, G.B.; Ramser, J.H. (1959). The Truck Dispatching Problem. *Management science*, vol 6, No 1 (Oct., 1959), pp. 80-91.

[7] De Koster, R., Le-Duc, T., & Roodbergen, K. J. (2007). Design and control of warehouse order picking: A literature review. *European Journal of Operational Research*, *182* (2), 481–501. doi:10.1016/j.ejor.2006.07.009

[8] Garey, M.R. (1976). The Complexity of Flowshop and Jopshop Scheduling. *Mathematics of Operations Research* 1 (2): 117-129. Doi:10.1287/moor.1.2.117.JSTOR 3689278

[9] Giffler B and Thompson G. (1959) Algorithms for solving production scheduling problems, IBMC. *International business machines corporation.*

[10] Johnson S.M. (1953) Optimal two- and three-stage production schedules with setup times included.

[11] Manikas, A., & Chang, Y.-L. (2009). Multi-criteria sequence-dependent job shop scheduling using genetic algorithms. *Computers & Industrial Engineering*, *56* (1), 179–185. doi:10.1016/j.cie.2008.05.002

[12] Manzini, R., Gamberi, M., & Regattieri, A. (2005). Design and control of an AS/RS. *The International Journal of Advanced Manufacturing Technology*, *28* (7-8), 766–774. doi:10.1007/s00170-004-2427-6

[13] Matković P. (2010). A Comparative Overview of the Evolution of Software Development Models, *1* (4), 163–172.

[14] Nanvala, H. (2011). Approaches for solving machine loading problem in FMS : A Review, *3* (May), 64–73.

[15] Oliveira, J. A. A Genetic Algorithm with a Quasi-local Search for the Job Shop Problem with Recirculation Classical Job Shop and Job Shop with Recirculation.

[16] Oliveira, J. A. (2007). Scheduling the truckload operations in automatic warehouses. *European Journal of Operational Research*, *179* (3), 723–735. doi:10.1016/j.ejor.2005.03.066

[17] OSQA (2009). SDLC Models.

[18] Papadimitriou, C.H. and Kanellakis, P.C., 1980. Flowshop scheduling with limited temporary storage. J. Assoc. Comput. Mach., 27: 533-549.

[19] Problem, P. F., & From, N. (n.d.). Flowshop Scheduling Problem Introduction Permutation Flowshop Scheduling Problem – Definition.

[20] Reddi, S. S. and Ramamoorthy C. V. (1972) On the flow-shop sequencing problem with no wait in process. Pergamon press, vol. 23, pp. 323-331.

[21] Rock, H., 1984. The three-machine no-wait flow-shop is NP-complete. J. Assoc. Comput. Mach., 31: 336-345.

[22] Sargent R.G. (2004) Verification and validation of simulation models, Syracuse, New York

[23] Schlesinger, S., et al. (1979), Terminology for the Model Credibility, Simualtion 32, 3 (Mar.), 103-104.

[24] Box G. E. P. and Hunter W. G., Statistics for Experimenters: An Introduction to Design, Data Analysis, and Model Building (New York: Wiley, 1978).

[24] Slomp, J., & Gupta, J. N. D. (1992). Interactive tool for scheduling jobs in a flexible manufacturing environment. *Computer Integrated Manufacturing Systems*, *5* (4), 291–299. doi:10.1016/0951-5240(92)90046-F

[25] Sviridenko, M. I. (2004). A Note on Permutation Flow Shop Problem. *Annals of Operations Research*, *129* (1-4), 247–252. doi:10.1023/B:ANOR.0000030691.65576.28

[26] Tulp J.J. (1974) Hoogbouwveem te Roosendaal

Appendix A

Background information on ORTEC

Since the foundation of ORTEC on the first of April 1981, the company has grown to be the market leader in advanced planning, optimization and logistic consultancy services. In the first year ORTEC started with 4 students of the study econometrics at the Erasmus University of Rotterdam, nowadays the company consist of over 700 employees on different locations in Europe as well as offices in the United States, North-America and South East Asia.[22] With more than 1650 customers all over the world, ORTEC is one of the largest providers of advanced software solutions and consultancy services for planning and optimization. All these different customers use ORTEC's solutions for different purposes as: demand forecasting, network/transport planning, loading vehicles and scheduling of services and timetables. These different solutions are mostly sold to customers in the way of standardized products, which are easily adaptable for each specific customer.

In 2010 within the compartment ORTEC Business Development (OBD) a group was formed that specializes in warehousing. More specific, the warehousing group wants to transform the old warehouses with fishbone structures and pallet-trucks driven by personnel, to fully automated warehouses with extensive complex lay-outs and many functions:

- Buffering ingoing and outgoing pallets to reduce (un-)loading times of trailers.
- Preparing trailer loads by combining orders of pallets, and sorting to customer specific rules.
- Keeping buffers replenished.
- Multiple entry and exit points: connect the warehouse to restacking areas, order picking areas, incoming and outgoing trucks.

The first project this warehousing group worked on was a warehouse for logistics service provider 'Van Heezik' in Spijkenisse[23] . This system did not work as it was intended to from the start, but from this learning process a new and better working warehouse was built for 'Van Heezik' at Lage Weide[Bron]. This was a comparable system as the Optilogx box constructed in the current projects, with the differences that not every buffer lane had its own motor. The movement of the buffer lanes was done by making use of the motor on the shuttle. Also the control was done differently than in the current projects. From this point on, the automated warehouses for FreezeServe and Hellendoorn were developed. These two projects are now running, but the final delivery still hasn´t taken place. The main reason for this, are the problems these warehouses are still coping with. ORTEC and the other companies involved are focusing to solve all these remaining issues in order to do the final delivery somewhere in the near future. The

[22] http://www.ortec.nl
[23] http://www.heezik.nl

warehousing division will then start to fully focus on the development of the warehouse in Northern France. The prospect for the developed warehousing software is that it will become one of ORTEC ´s standardized products to sell to customers.

Appendix B

This appendix will present two figures in which the dig-out times from and to every lane per level is presented in figure 1 and the Outfeed routes and processtimes from every lane to the MUP is presented in figure 2. Both these figures are presented on the next pages.

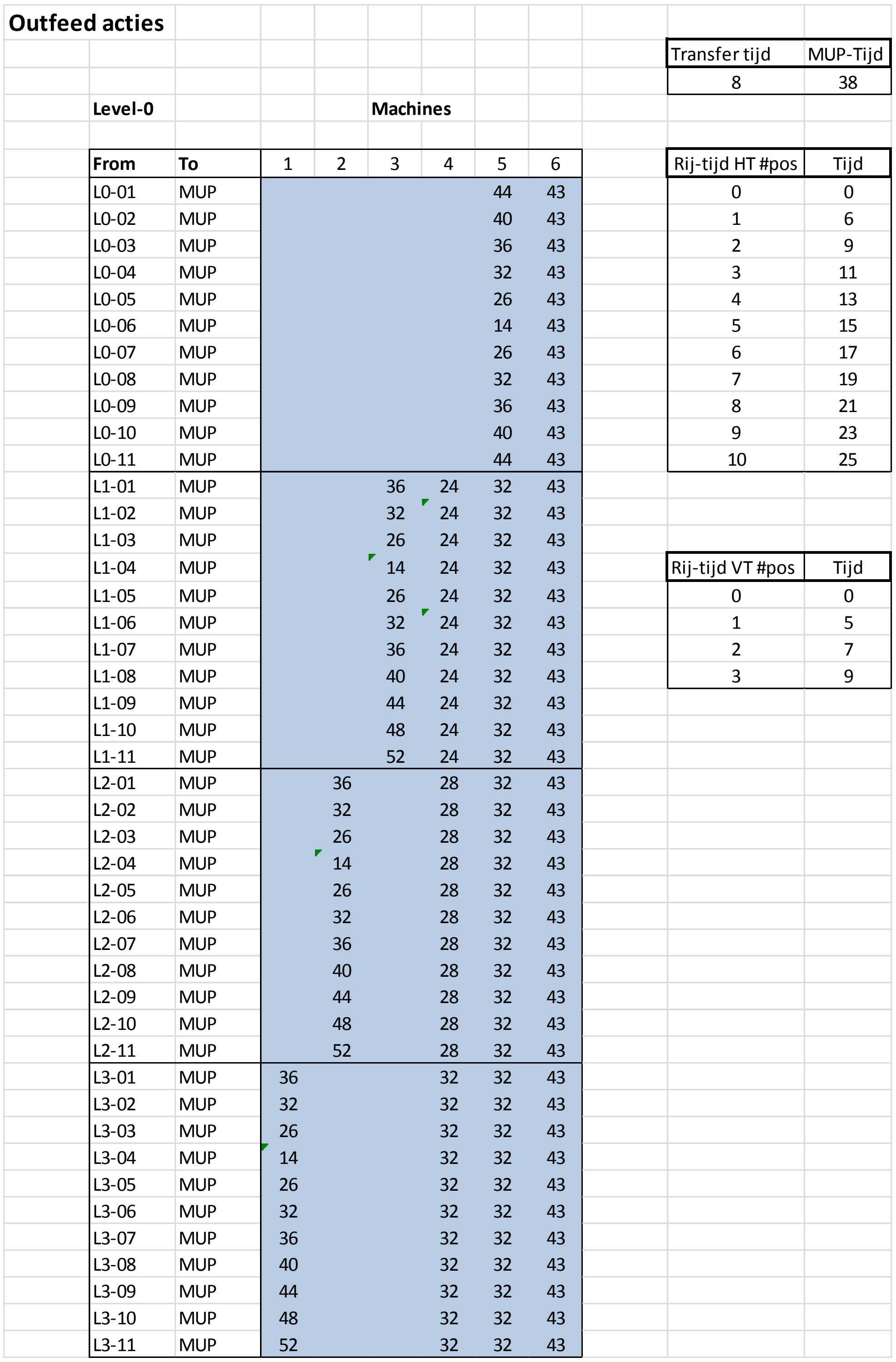
Level-0				Machines			
From	To	1	2	3	4	5	6
L0-01	L0-01	-	-	-	-	-	-
L0-01	L0-02					20	
L0-01	L0-03					23	
L0-01	L0-04					25	
L0-01	L0-05					27	
L0-01	L0-06					29	
L0-01	L0-07					31	
L0-01	L0-08					33	
L0-01	L0-09					35	
L0-01	L0-10					37	
L0-01	L0-11					39	
L0-02	L0-01					20	
L0-02	L0-02	-	-	-	-	-	-
L0-02	L0-03					20	
L0-02	L0-04					23	
L0-02	L0-05					25	
L0-02	L0-06					27	
L0-02	L0-07					29	
L0-02	L0-08					31	
L0-02	L0-09					33	
L0-02	L0-10					35	
L0-02	L0-11					37	
L0-03	L0-01					23	
L0-03	L0-02					20	
L0-03	L0-03	-	-	-	-	-	-
L0-03	L0-04					20	
L0-03	L0-05					23	
L0-03	L0-06					25	
L0-03	L0-07					27	
L0-03	L0-08					29	
L0-03	L0-09					31	
L0-03	L0-10					33	
L0-03	L0-11					35	
L0-04	L0-01					25	
L0-04	L0-02					23	
L0-04	L0-03					20	
L0-04	L0-04	-	-	-	-	-	-
L0-04	L0-05					20	
L0-04	L0-06					23	
L0-04	L0-07					25	
L0-04	L0-08					27	
L0-04	L0-09					29	
L0-04	L0-10					31	
L0-04	L0-11					33	
L0-05	L0-01					27	
L0-05	L0-02					25	
L0-05	L0-03					23	
L0-05	L0-04					20	
L0-05	L0-05	-	-	-	-	-	-
L0-05	L0-06					20	
L0-05	L0-07					25	
L0-05	L0-08					25	
L0-05	L0-09					27	
L0-05	L0-10					29	
L0-05	L0-11					31	
L0-06	L0-01					29	
L0-06	L0-02					27	
L0-06	L0-03					25	
L0-06	L0-04					23	
L0-06	L0-05					20	
L0-06	L0-06	-	-	-	-	-	-
L0-06	L0-07					20	
L0-06	L0-08					23	
L0-06	L0-09					25	
L0-06	L0-10					27	
L0-06	L0-11					29	
L0-07	L0-01					31	
L0-07	L0-02					29	
L0-07	L0-03					27	
L0-07	L0-04					25	
L0-07	L0-05					23	
L0-07	L0-06					20	
L0-07	L0-07	-	-	-	-	-	-
L0-07	L0-08					20	
L0-07	L0-09					23	
L0-07	L0-10					25	
L0-07	L0-11					27	
L0-08	L0-01					33	
L0-08	L0-02					31	
L0-08	L0-03					29	
L0-08	L0-04					27	
L0-08	L0-05					25	
L0-08	L0-06					23	
L0-08	L0-07					20	
L0-08	L0-08	-	-	-	-	-	-
L0-08	L0-09					20	
L0-08	L0-10					23	
L0-08	L0-11					25	
L0-09	L0-01					35	
L0-09	L0-02					33	
L0-09	L0-03					31	
L0-09	L0-04					29	
L0-09	L0-05					27	
L0-09	L0-06					25	
L0-09	L0-07					23	
L0-09	L0-08					20	
L0-09	L0-09	-	-	-	-	-	-
L0-09	L0-10					22	
L0-09	L0-11					23	
L0-10	L0-01					37	
L0-10	L0-02					35	
L0-10	L0-03					33	
L0-10	L0-04					31	
L0-10	L0-05					29	
L0-10	L0-06					27	
L0-10	L0-07					25	
L0-10	L0-08					23	
L0-10	L0-09					20	
L0-10	L0-10	-	-	-	-	-	-
L0-10	L0-11					20	
L0-11	L0-01					39	
L0-11	L0-02					37	
L0-11	L0-03					35	
L0-11	L0-04					33	
L0-11	L0-05					31	
L0-11	L0-06					29	
L0-11	L0-07					27	
L0-11	L0-08					25	
L0-11	L0-09					23	
L0-11	L0-10					20	
L0-11	L0-11	-	-	-	-	-	-

Level-1				Machines			
From	To	1	2	3	4	5	6
L1-01	L1-01	-	-	-	-	-	-
L1-01	L1-02			20			
L1-01	L1-03			23			
L1-01	L1-04			25			
L1-01	L1-05			27			
L1-01	L1-06			29			
L1-01	L1-07			31			
L1-01	L1-08			33			
L1-01	L1-09			35			
L1-01	L1-10			37			
L1-01	L1-11			39			
L1-02	L1-01			20			
L1-02	L1-02	-	-	-	-	-	-
L1-02	L1-03			20			
L1-02	L1-04			23			
L1-02	L1-05			25			
L1-02	L1-06			27			
L1-02	L1-07			29			
L1-02	L1-08			31			
L1-02	L1-09			33			
L1-02	L1-10			35			
L1-02	L1-11			37			
L1-03	L1-01			23			
L1-03	L1-02			20			
L1-03	L1-03	-	-	-	-	-	-
L1-03	L1-04			20			
L1-03	L1-05			23			
L1-03	L1-06			25			
L1-03	L1-07			27			
L1-03	L1-08			29			
L1-03	L1-09			31			
L1-03	L1-10			33			
L1-03	L1-11			35			
L1-04	L1-01			25			
L1-04	L1-02			23			
L1-04	L1-03			20			
L1-04	L1-04	-	-	-	-	-	-
L1-04	L1-05			20			
L1-04	L1-06			23			
L1-04	L1-07			25			
L1-04	L1-08			27			
L1-04	L1-09			29			
L1-04	L1-10			31			
L1-04	L1-11			33			
L1-05	L1-01			27			
L1-05	L1-02			25			
L1-05	L1-03			23			
L1-05	L1-04			20			
L1-05	L1-05	-	-	-	-	-	-
L1-05	L1-06			20			
L1-05	L1-07			25			
L1-05	L1-08			25			
L1-05	L1-09			27			
L1-05	L1-10			29			
L1-05	L1-11			31			
L1-06	L1-01			29			
L1-06	L1-02			27			
L1-06	L1-03			25			
L1-06	L1-04			23			
L1-06	L1-05			20			
L1-06	L1-06	-	-	-	-	-	-
L1-06	L1-07			20			
L1-06	L1-08			23			
L1-06	L1-09			25			
L1-06	L1-10			27			
L1-06	L1-11			29			
L1-07	L1-01			31			
L1-07	L1-02			29			
L1-07	L1-03			27			
L1-07	L1-04			25			
L1-07	L1-05			23			
L1-07	L1-06			20			
L1-07	L1-07	-	-	-	-	-	-
L1-07	L1-08			20			
L1-07	L1-09			23			
L1-07	L1-10			25			
L1-07	L1-11			27			
L1-08	L1-01			33			
L1-08	L1-02			31			
L1-08	L1-03			29			
L1-08	L1-04			27			
L1-08	L1-05			25			
L1-08	L1-06			23			
L1-08	L1-07			20			
L1-08	L1-08	-	-	-	-	-	-
L1-08	L1-09			20			
L1-08	L1-10			23			
L1-08	L1-11			25			
L1-09	L1-01			35			
L1-09	L1-02			33			
L1-09	L1-03			31			
L1-09	L1-04			29			
L1-09	L1-05			27			
L1-09	L1-06			25			
L1-09	L1-07			23			
L1-09	L1-08			20			
L1-09	L1-09	-	-	-	-	-	-
L1-09	L1-10			22			
L1-09	L1-11			23			
L1-10	L1-01			37			
L1-10	L1-02			35			
L1-10	L1-03			33			
L1-10	L1-04			31			
L1-10	L1-05			29			
L1-10	L1-06			27			
L1-10	L1-07			25			
L1-10	L1-08			23			
L1-10	L1-09			20			
L1-10	L1-10	-	-	-	-	-	-
L1-10	L1-11			20			
L1-11	L1-01			39			
L1-11	L1-02			37			
L1-11	L1-03			35			
L1-11	L1-04			33			
L1-11	L1-05			31			
L1-11	L1-06			29			
L1-11	L1-07			27			
L1-11	L1-08			25			
L1-11	L1-09			23			
L1-11	L1-10			20			
L1-11	L1-11	-	-	-	-	-	-

Level-2				Machines			
From	To	1	2	3	4	5	6
L2-01	L2-01	-	-	-	-	-	-
L2-01	L2-02		20				
L2-01	L2-03		23				
L2-01	L2-04		25				
L2-01	L2-05		27				
L2-01	L2-06		29				
L2-01	L2-07		31				
L2-01	L2-08		33				
L2-01	L2-09		35				
L2-01	L2-10		37				
L2-01	L2-11		39				
L2-02	L2-01		20				
L2-02	L2-02	-	-	-	-	-	-
L2-02	L2-03		20				
L2-02	L2-04		23				
L2-02	L2-05		25				
L2-02	L2-06		27				
L2-02	L2-07		29				
L2-02	L2-08		31				
L2-02	L2-09		33				
L2-02	L2-10		35				
L2-02	L2-11		37				
L2-03	L2-01		23				
L2-03	L2-02		20				
L2-03	L2-03	-	-	-	-	-	-
L2-03	L2-04		20				
L2-03	L2-05		23				
L2-03	L2-06		25				
L2-03	L2-07		27				
L2-03	L2-08		29				
L2-03	L2-09		31				
L2-03	L2-10		33				
L2-03	L2-11		35				
L2-04	L2-01		25				
L2-04	L2-02		23				
L2-04	L2-03		20				
L2-04	L2-04	-	-	-	-	-	-
L2-04	L2-05		20				
L2-04	L2-06		23				
L2-04	L2-07		25				
L2-04	L2-08		27				
L2-04	L2-09		29				
L2-04	L2-10		31				
L2-04	L2-11		33				
L2-05	L2-01		27				
L2-05	L2-02		25				
L2-05	L2-03		23				
L2-05	L2-04		20				
L2-05	L2-05	-	-	-	-	-	-
L2-05	L2-06		20				
L2-05	L2-07		25				
L2-05	L2-08		25				
L2-05	L2-09		27				
L2-05	L2-10		29				
L2-05	L2-11		31				
L2-06	L2-01		29				
L2-06	L2-02		27				
L2-06	L2-03		25				
L2-06	L2-04		23				
L2-06	L2-05		20				
L2-06	L2-06	-	-	-	-	-	-
L2-06	L2-07		20				
L2-06	L2-08		23				
L2-06	L2-09		25				
L2-06	L2-10		27				
L2-06	L2-11		29				
L2-07	L2-01		31				
L2-07	L2-02		29				
L2-07	L2-03		27				
L2-07	L2-04		25				
L2-07	L2-05		23				
L2-07	L2-06		20				
L2-07	L2-07	-	-	-	-	-	-
L2-07	L2-08		20				
L2-07	L2-09		23				
L2-07	L2-10		25				
L2-07	L2-11		27				
L2-08	L2-01		33				
L2-08	L2-02		31				
L2-08	L2-03		29				
L2-08	L2-04		27				
L2-08	L2-05		25				
L2-08	L2-06		23				
L2-08	L2-07		20				
L2-08	L2-08	-	-	-	-	-	-
L2-08	L2-09		20				
L2-08	L2-10		23				
L2-08	L2-11		25				
L2-09	L2-01		35				
L2-09	L2-02		33				
L2-09	L2-03		31				
L2-09	L2-04		29				
L2-09	L2-05		27				
L2-09	L2-06		25				
L2-09	L2-07		23				
L2-09	L2-08		20				
L2-09	L2-09	-	-	-	-	-	-
L2-09	L2-10		22				
L2-09	L2-11		23				
L2-10	L2-01		37				
L2-10	L2-02		35				
L2-10	L2-03		33				
L2-10	L2-04		31				
L2-10	L2-05		29				
L2-10	L2-06		27				
L2-10	L2-07		25				
L2-10	L2-08		23				
L2-10	L2-09		20				
L2-10	L2-10	-	-	-	-	-	-
L2-10	L2-11		20				
L2-11	L2-01		39				
L2-11	L2-02		37				
L2-11	L2-03		35				
L2-11	L2-04		33				
L2-11	L2-05		31				
L2-11	L2-06		29				
L2-11	L2-07		27				
L2-11	L2-08		25				
L2-11	L2-09		23				
L2-11	L2-10		20				
L2-11	L2-11	-	-	-	-	-	-

Level-3				Machines			
From	To	1	2	3	4	5	6
L3-01	L3-01	-	-	-	-	-	-
L3-01	L3-02	20					
L3-01	L3-03	23					
L3-01	L3-04	25					
L3-01	L3-05	27					
L3-01	L3-06	29					
L3-01	L3-07	31					
L3-01	L3-08	33					
L3-01	L3-09	35					
L3-01	L3-10	37					
L3-01	L3-11	39					
L3-02	L3-01	20					
L3-02	L3-02	-	-	-	-	-	-
L3-02	L3-03	20					
L3-02	L3-04	23					
L3-02	L3-05	25					
L3-02	L3-06	27					
L3-02	L3-07	29					
L3-02	L3-08	31					
L3-02	L3-09	33					
L3-02	L3-10	35					
L3-02	L3-11	37					
L3-03	L3-01	23					
L3-03	L3-02	20					
L3-03	L3-03	-	-	-	-	-	-
L3-03	L3-04	20					
L3-03	L3-05	23					
L3-03	L3-06	25					
L3-03	L3-07	27					
L3-03	L3-08	29					
L3-03	L3-09	31					
L3-03	L3-10	33					
L3-03	L3-11	35					
L3-04	L3-01	25					
L3-04	L3-02	23					
L3-04	L3-03	20					
L3-04	L3-04	-	-	-	-	-	-
L3-04	L3-05	20					
L3-04	L3-06	23					
L3-04	L3-07	25					
L3-04	L3-08	27					
L3-04	L3-09	29					
L3-04	L3-10	31					
L3-04	L3-11	33					
L3-05	L3-01	27					
L3-05	L3-02	25					
L3-05	L3-03	23					
L3-05	L3-04	20					
L3-05	L3-05	-	-	-	-	-	-
L3-05	L3-06	20					
L3-05	L3-07	25					
L3-05	L3-08	25					
L3-05	L3-09	27					
L3-05	L3-10	29					
L3-05	L3-11	31					
L3-06	L3-01	29					
L3-06	L3-02	27					
L3-06	L3-03	25					
L3-06	L3-04	23					
L3-06	L3-05	20					
L3-06	L3-06	-	-	-	-	-	-
L3-06	L3-07	20					
L3-06	L3-08	23					
L3-06	L3-09	25					
L3-06	L3-10	27					
L3-06	L3-11	29					
L3-07	L3-01	31					
L3-07	L3-02	29					
L3-07	L3-03	27					
L3-07	L3-04	25					
L3-07	L3-05	23					
L3-07	L3-06	20					
L3-07	L3-07	-	-	-	-	-	-
L3-07	L3-08	20					
L3-07	L3-09	23					
L3-07	L3-10	25					
L3-07	L3-11	27					
L3-08	L3-01	33					
L3-08	L3-02	31					
L3-08	L3-03	29					
L3-08	L3-04	27					
L3-08	L3-05	25					
L3-08	L3-06	23					
L3-08	L3-07	20					
L3-08	L3-08	-	-	-	-	-	-
L3-08	L3-09	20					
L3-08	L3-10	23					
L3-08	L3-11	25					
L3-09	L3-01	35					
L3-09	L3-02	33					
L3-09	L3-03	31					
L3-09	L3-04	29					
L3-09	L3-05	27					
L3-09	L3-06	25					
L3-09	L3-07	23					
L3-09	L3-08	20					
L3-09	L3-09	-	-	-	-	-	-
L3-09	L3-10	22					
L3-09	L3-11	23					
L3-10	L3-01	37					
L3-10	L3-02	35					
L3-10	L3-03	33					
L3-10	L3-04	31					
L3-10	L3-05	29					
L3-10	L3-06	27					
L3-10	L3-07	25					
L3-10	L3-08	23					
L3-10	L3-09	20					
L3-10	L3-10	-	-	-	-	-	-
L3-10	L3-11	20					
L3-11	L3-01	39					
L3-11	L3-02	37					
L3-11	L3-03	35					
L3-11	L3-04	33					
L3-11	L3-05	31					
L3-11	L3-06	29					
L3-11	L3-07	27					
L3-11	L3-08	25					
L3-11	L3-09	23					
L3-11	L3-10	20					
L3-11	L3-11	-	-	-	-	-	-

Figure 1: Dig-out times



Outfeed acties

Transfer tijd	MUP-Tijd
8	38

Level-0 **Machines**

From	To	1	2	3	4	5	6
L0-01	MUP					44	43
L0-02	MUP					40	43
L0-03	MUP					36	43
L0-04	MUP					32	43
L0-05	MUP					26	43
L0-06	MUP					14	43
L0-07	MUP					26	43
L0-08	MUP					32	43
L0-09	MUP					36	43
L0-10	MUP					40	43
L0-11	MUP					44	43
L1-01	MUP			36	24	32	43
L1-02	MUP			32	24	32	43
L1-03	MUP			26	24	32	43
L1-04	MUP			14	24	32	43
L1-05	MUP			26	24	32	43
L1-06	MUP			32	24	32	43
L1-07	MUP			36	24	32	43
L1-08	MUP			40	24	32	43
L1-09	MUP			44	24	32	43
L1-10	MUP			48	24	32	43
L1-11	MUP			52	24	32	43
L2-01	MUP		36		28	32	43
L2-02	MUP		32		28	32	43
L2-03	MUP		26		28	32	43
L2-04	MUP		14		28	32	43
L2-05	MUP		26		28	32	43
L2-06	MUP		32		28	32	43
L2-07	MUP		36		28	32	43
L2-08	MUP		40		28	32	43
L2-09	MUP		44		28	32	43
L2-10	MUP		48		28	32	43
L2-11	MUP		52		28	32	43
L3-01	MUP	36			32	32	43
L3-02	MUP	32			32	32	43
L3-03	MUP	26			32	32	43
L3-04	MUP	14			32	32	43
L3-05	MUP	26			32	32	43
L3-06	MUP	32			32	32	43
L3-07	MUP	36			32	32	43
L3-08	MUP	40			32	32	43
L3-09	MUP	44			32	32	43
L3-10	MUP	48			32	32	43
L3-11	MUP	52			32	32	43

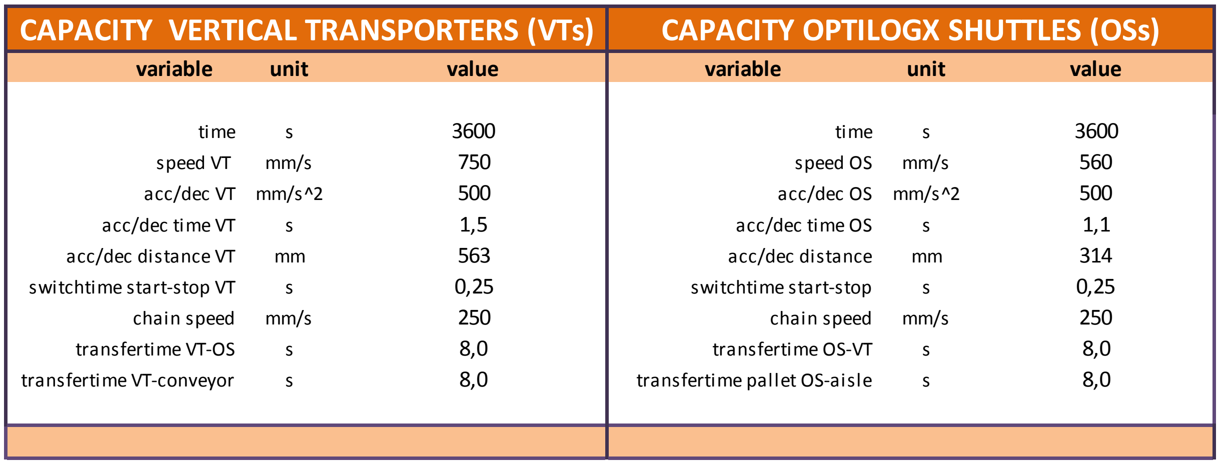
Rij-tijd HT #pos	Tijd
0	0
1	6
2	9
3	11
4	13
5	15
6	17
7	19
8	21
9	23
10	25

Rij-tijd VT #pos	Tijd
0	0
1	5
2	7
3	9

Figure 2: Outfeed action times

Appendix C

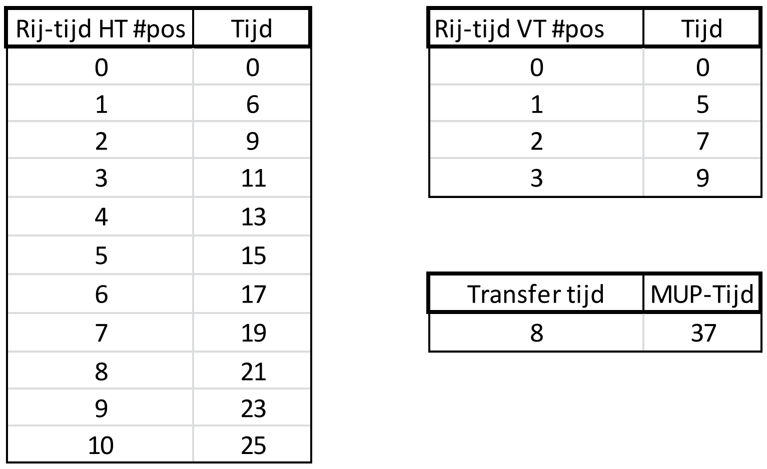
Calculation values for the Vertical Transporter capacity and the Optilogx shuttle capacity.



CAPACITY VERTICAL TRANSPORTERS (VTs)		
variable	unit	value
time	s	3600
speed VT	mm/s	750
acc/dec VT	mm/s^2	500
acc/dec time VT	s	1,5
acc/dec distance VT	mm	563
switchtime start-stop VT	s	0,25
chain speed	mm/s	250
transfertime VT-OS	s	8,0
transfertime VT-conveyor	s	8,0

CAPACITY OPTILOGX SHUTTLES (OSs)		
variable	unit	value
time	s	3600
speed OS	mm/s	560
acc/dec OS	mm/s^2	500
acc/dec time OS	s	1,1
acc/dec distance	mm	314
switchtime start-stop	s	0,25
chain speed	mm/s	250
transfertime OS-VT	s	8,0
transfertime pallet OS-aisle	s	8,0

Figure 1: Capacity VT and HT



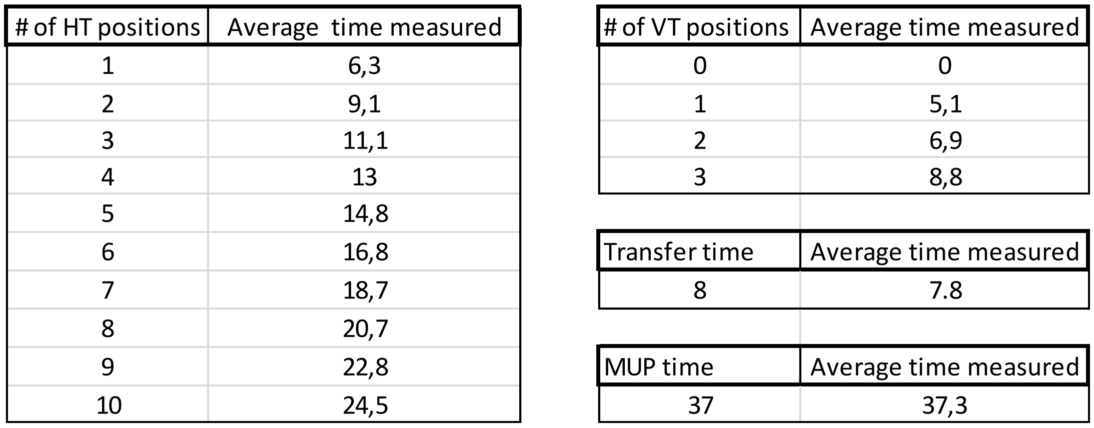
Rij-tijd HT #pos	Tijd
0	0
1	6
2	9
3	11
4	13
5	15
6	17
7	19
8	21
9	23
10	25

Rij-tijd VT #pos	Tijd
0	0
1	5
2	7
3	9

Transfer tijd	MUP-Tijd
8	37

Figure Error! Use the Home tab to apply 0 to the text that you want to appear here.**4: Times HT and VT positions, and transfer/MUP times**

Average measured times for the Horizontal Transporter positions, the Vertical Transporter positions, the transfer times and the MUP time.



# of HT positions	Average time measured
1	6,3
2	9,1
3	11,1
4	13
5	14,8
6	16,8
7	18,7
8	20,7
9	22,8
10	24,5

# of VT positions	Average time measured
0	0
1	5,1
2	6,9
3	8,8

Transfer time	Average time measured
8	7.8

MUP time	Average time measured
37	37,3

Figure 5: Average measured times for HT, VT transfer times and MUP time

Appendix D

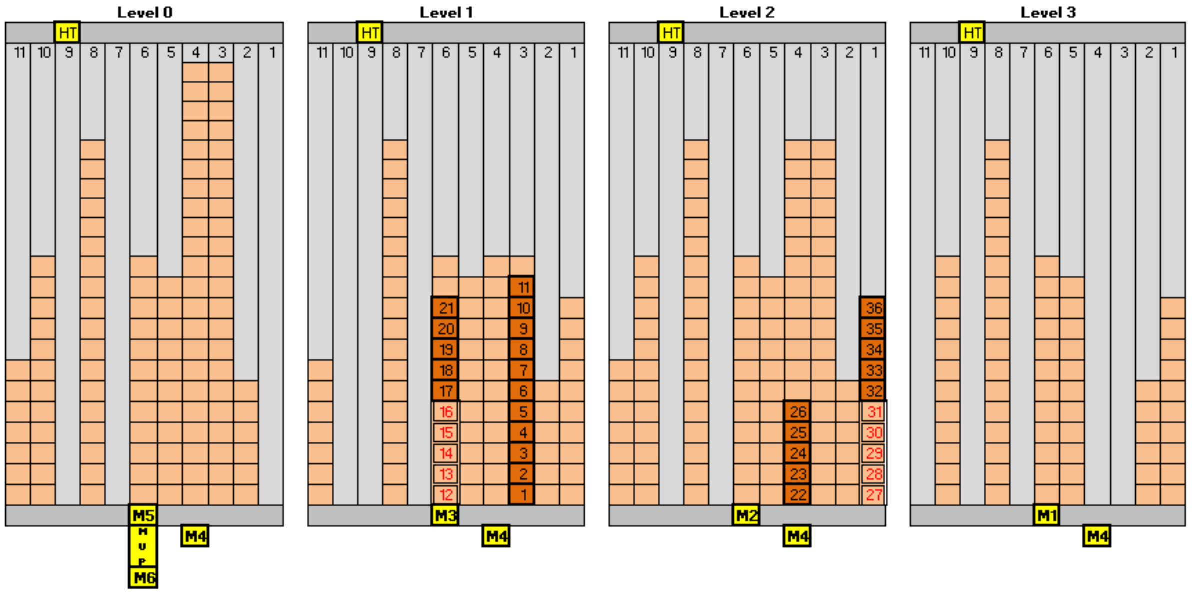
The tests are performed on a machine with the following configuration:

- Intel(R) Core(TM) I5-3320M CPU @2.60GHz
- 8 GB RAM

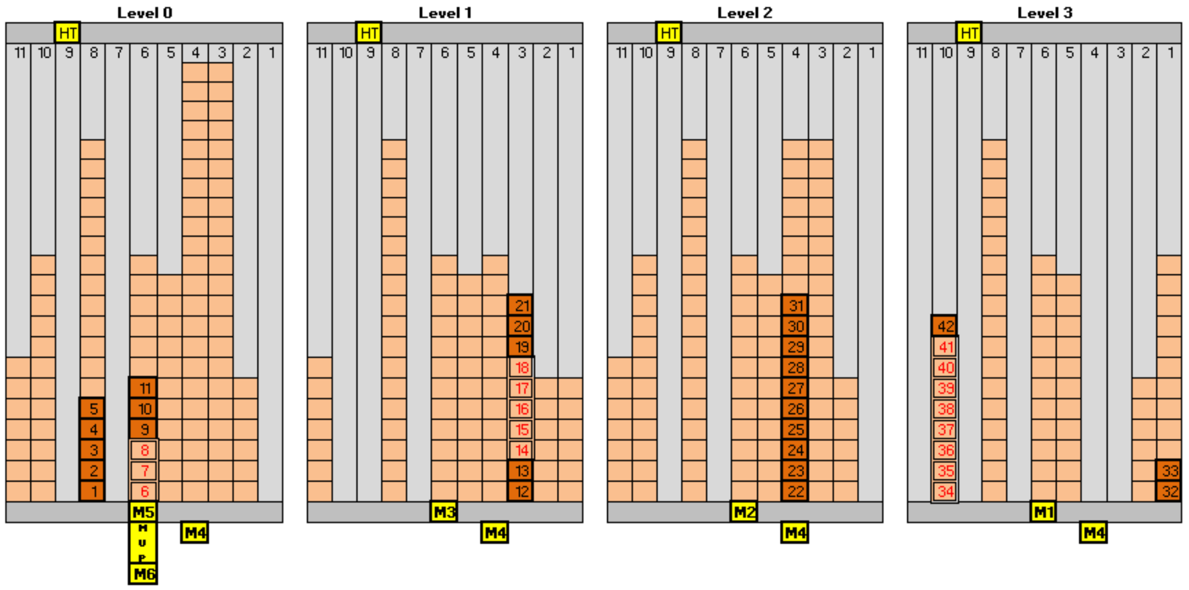
Appendix E

Cases:

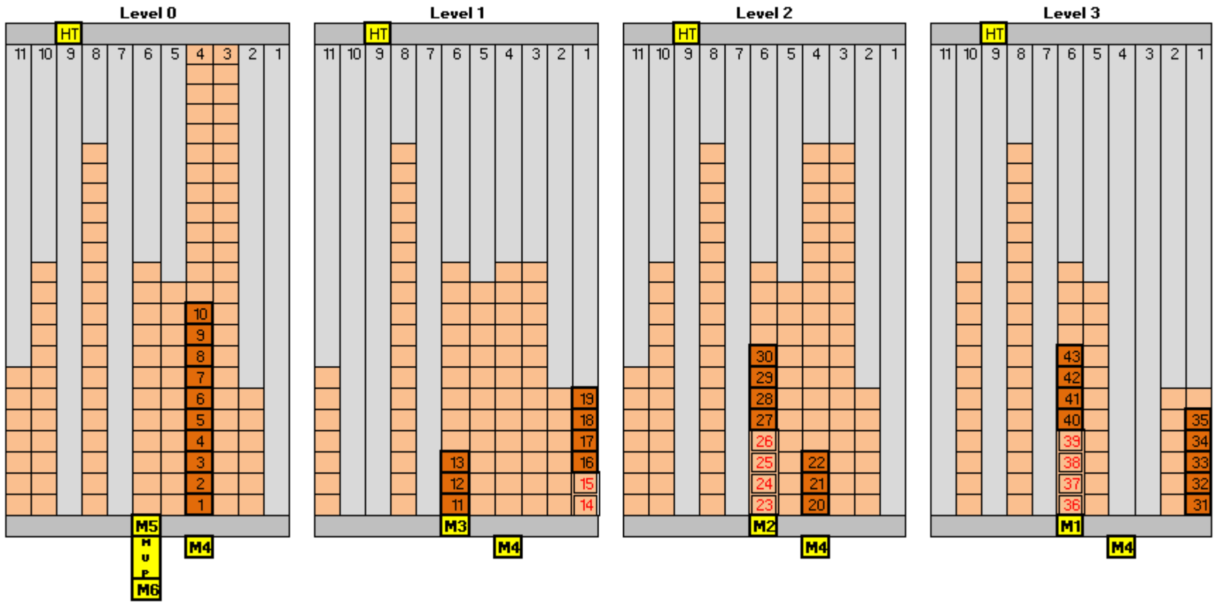
Case 1:



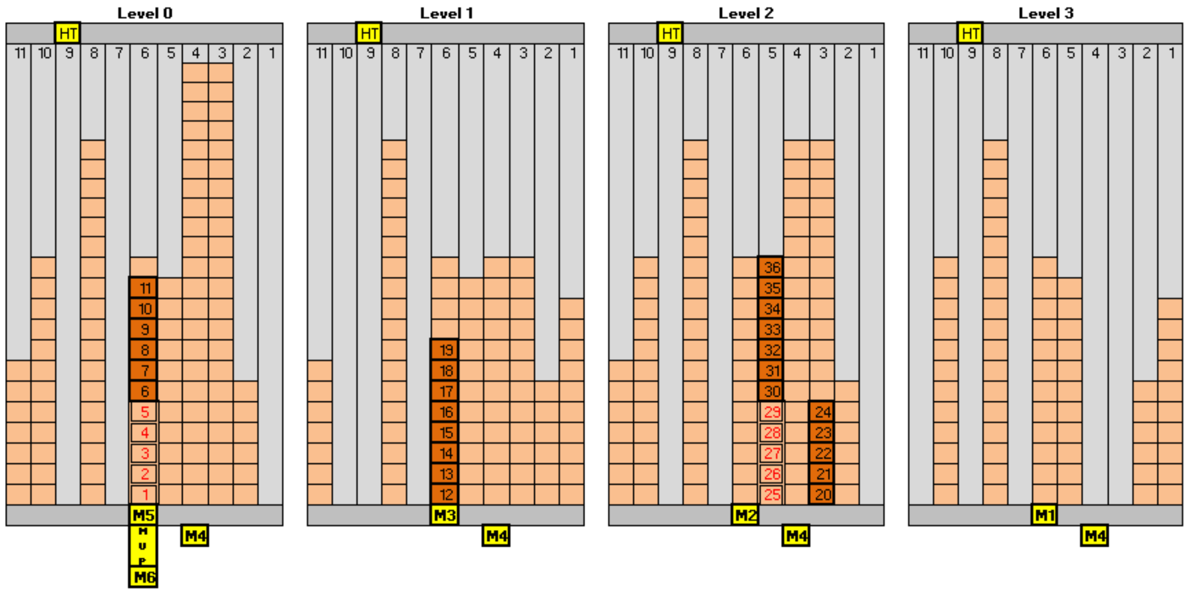
Case 2:



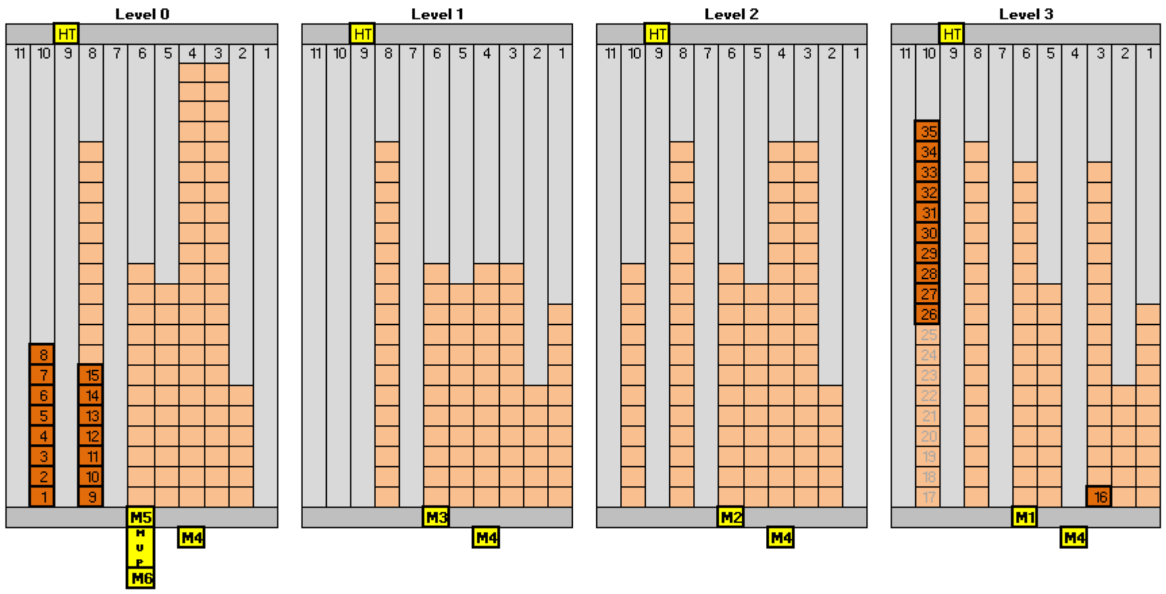
Case 3:


Level 0
Level 1
Level 2
Level 3
HT
11 10 9 8 7 6 5 4 3 2 1
M5
M6
M4
M3
M2
M1

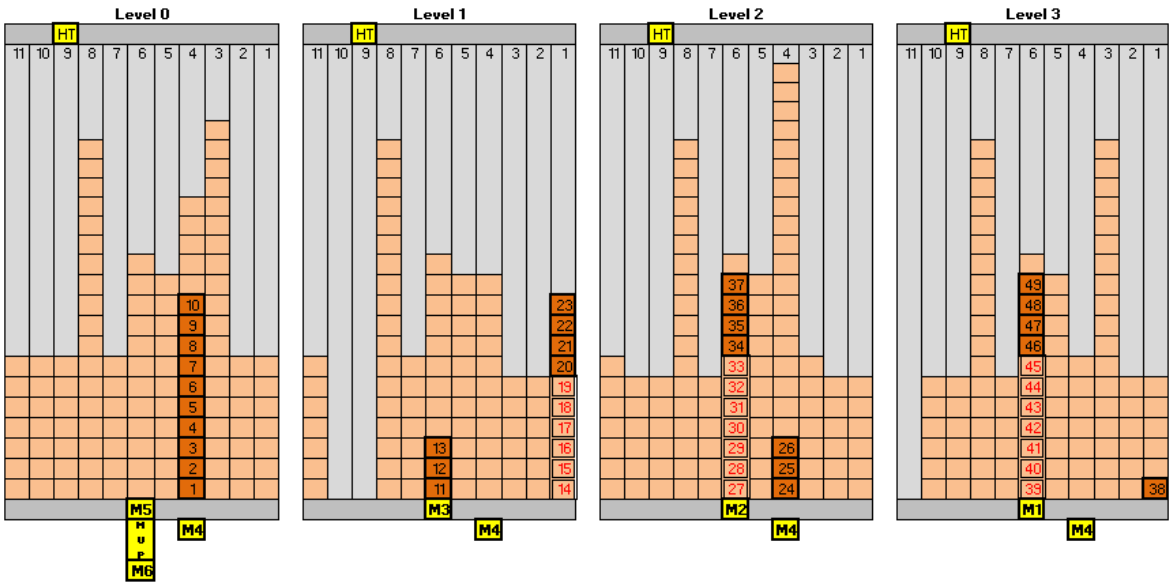
Case 4:


Level 0
Level 1
Level 2
Level 3
HT
11 10 9 8 7 6 5 4 3 2 1
M5
M6
M4
M3
M2
M1

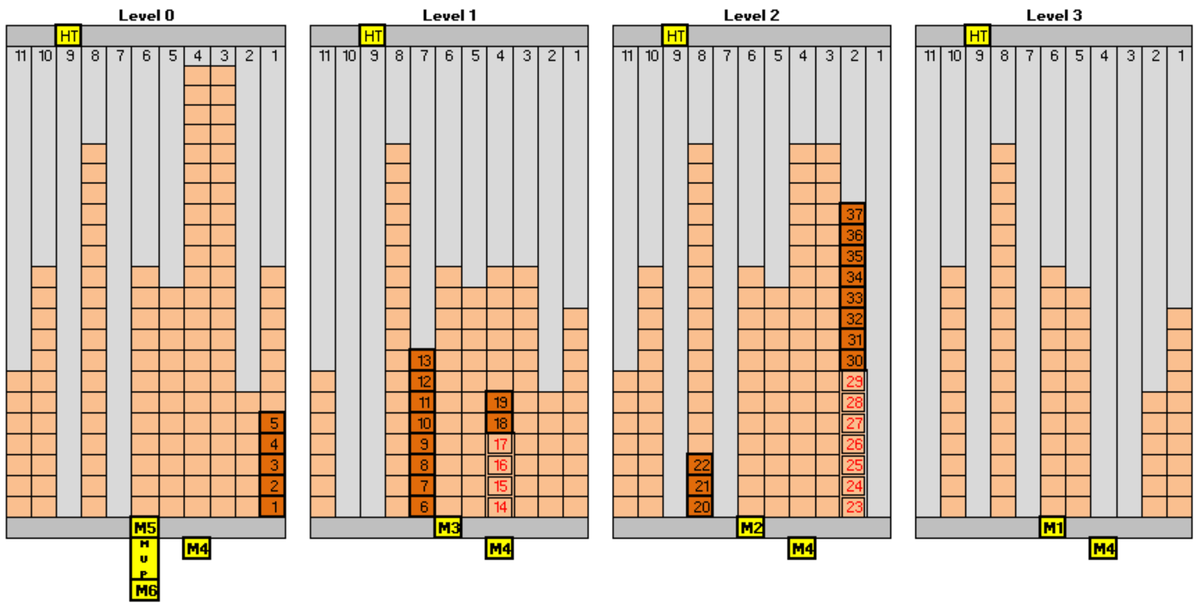
Case 5:


Level 0
Level 1
Level 2
Level 3
HT
11 10 9 8 7 6 5 4 3 2 1
M5
M6
M4
M3
M2
M1
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 33 34 35

Case 6:

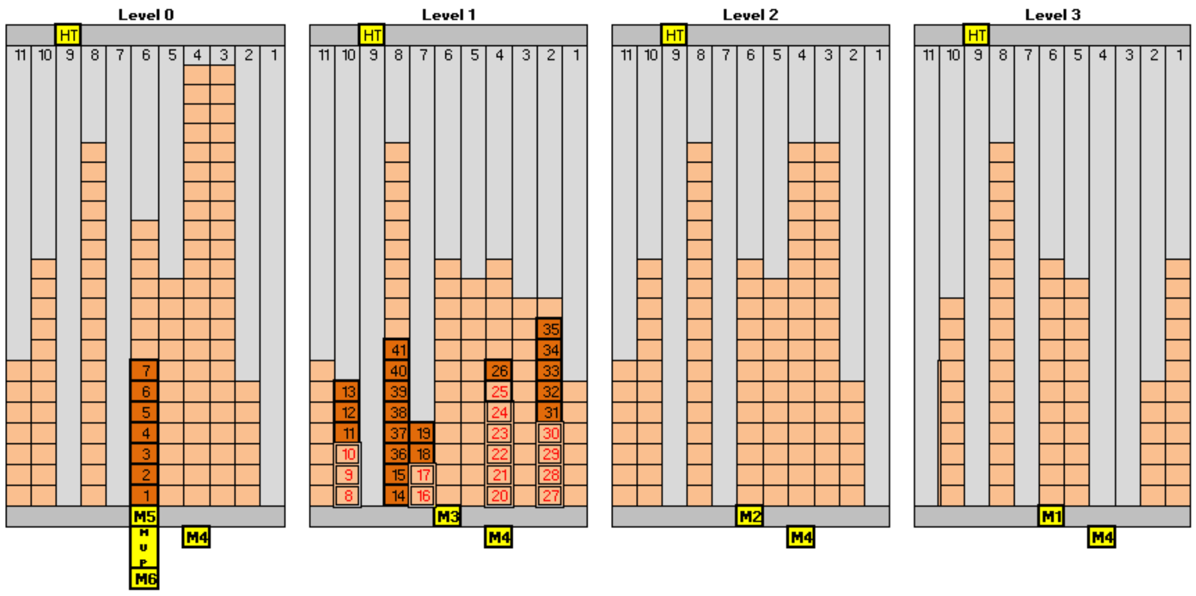

Level 0
Level 1
Level 2
Level 3
HT
11 10 9 8 7 6 5 4 3 2 1
M5
M6
M4
M3
M2
M1
1 2 3 4 5 6 7 8 9 10
11 12 13
14 15 16 17 18 19 20 21 22 23
24 25 26
27 28 29 30 31 32 33 34 35 36 37
38
39 40 41 42 43 44 45 46 47 48 49

Case 7:



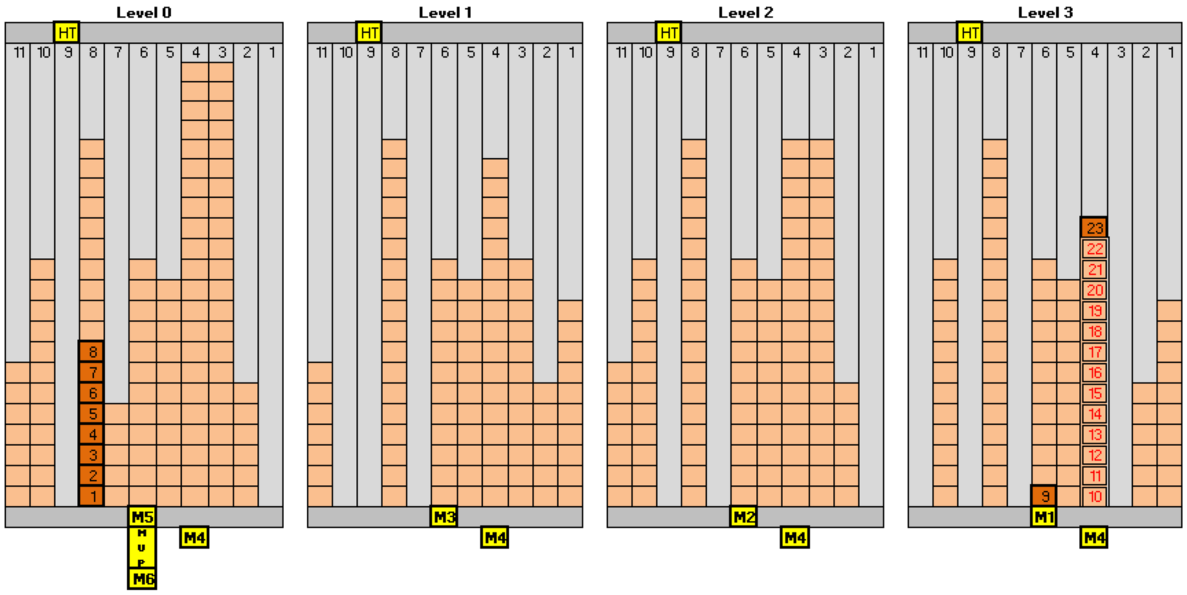
Level 0
Level 1
Level 2
Level 3
HT
M1
M2
M3
M4
M5
M6

Case 8:

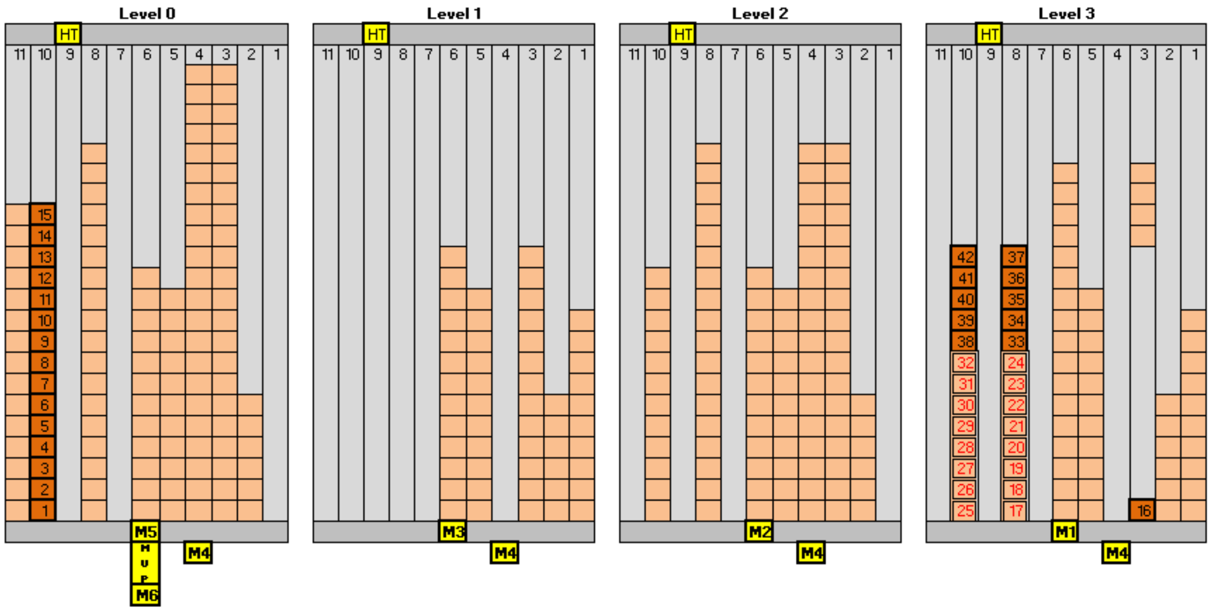


Level 0
Level 1
Level 2
Level 3
HT
M1
M2
M3
M4
M5
M6

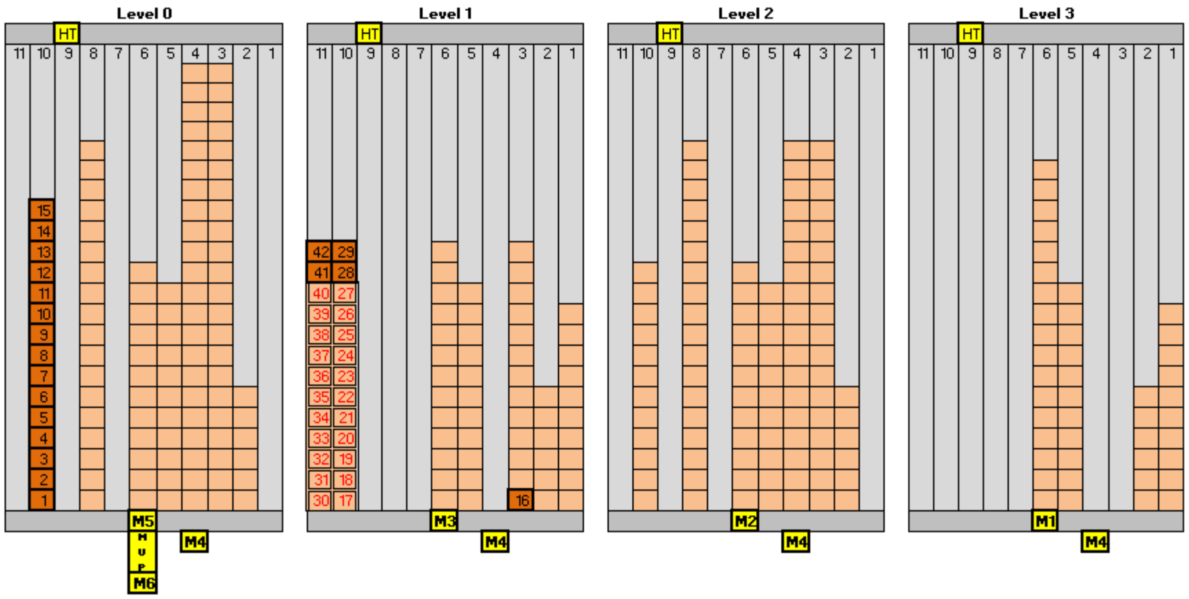
Case 9:



Case 10:

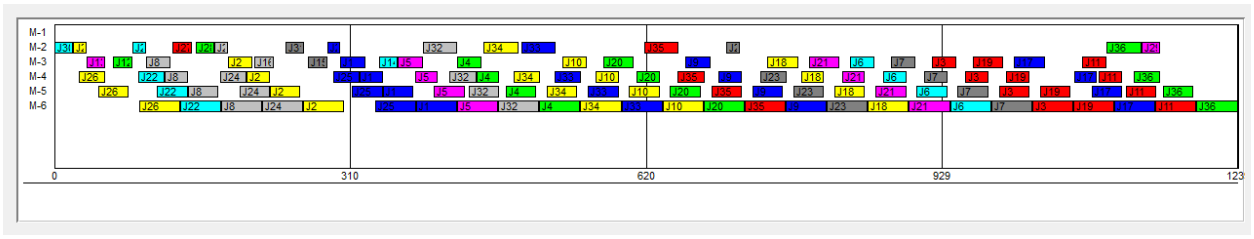


Case 11:



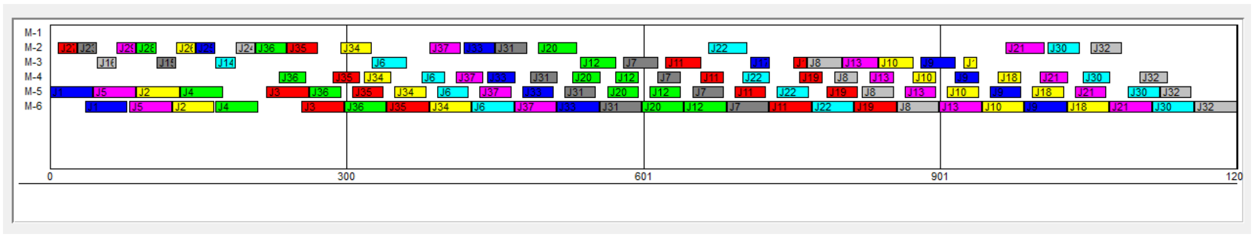
Appendix F

Case: 1



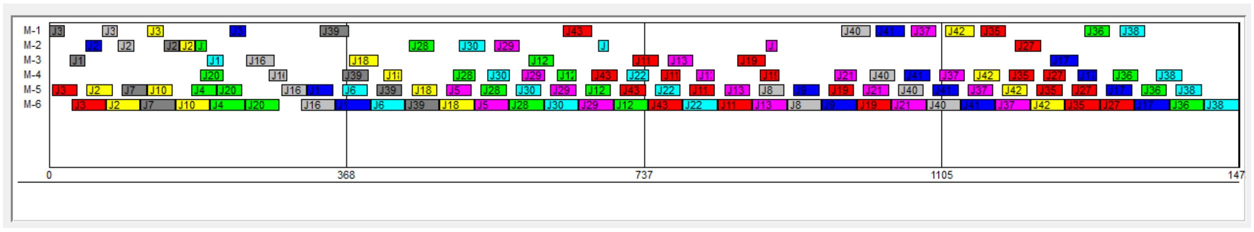
Simulation Results [seconds]		
10 Sec_MIP Model		1239
MIP Model Optimization	=	1168
Model Calculation Time	=	10
Emulation Old	=	**1402**
Emulation MIP	=	**1197**

Case: 2



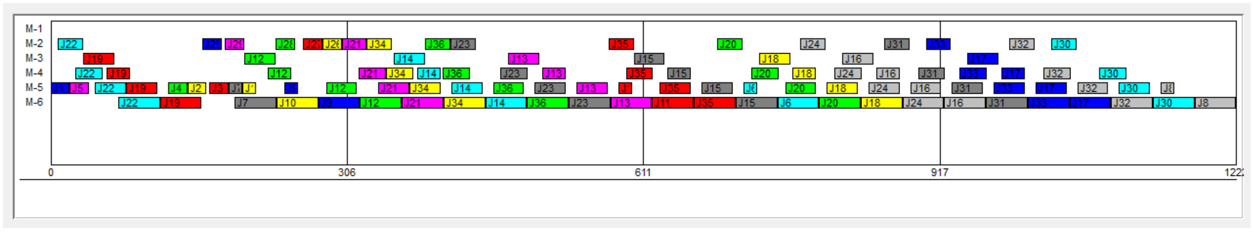
Simulation Results [seconds]		
10 Sec_MIP Model		1181
MIP Model Optimization	=	1167
Model Calculation Time	=	10
Emulation Old	=	**1400**
Emulation MIP	=	**1168**

Case: 3



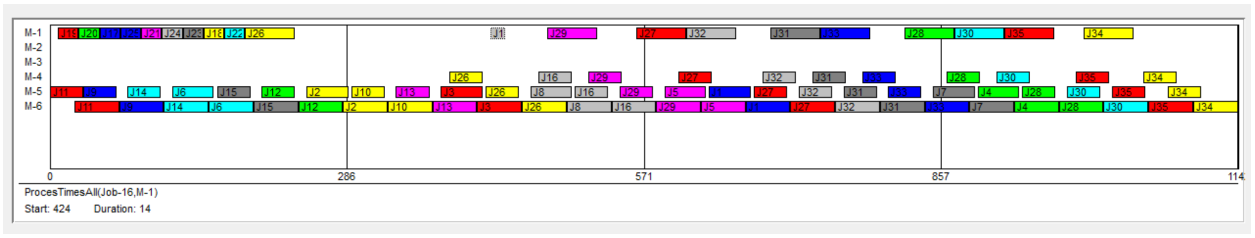
Simulation Results [seconds]		
10 Sec_MIP Model		1473
MIP Model Optimization	=	1443
Model Calculation Time	=	10
Emulation Old	=	**1672**
Emulation MIP	=	**1443**

Case: 4



Simulation Results [seconds]		
10 Sec_MIP Model		1222
MIP Model Optimization	=	1202
Model Calculation Time	=	10
Emulation Old	=	**1340**
Emulation MIP	=	**1220**

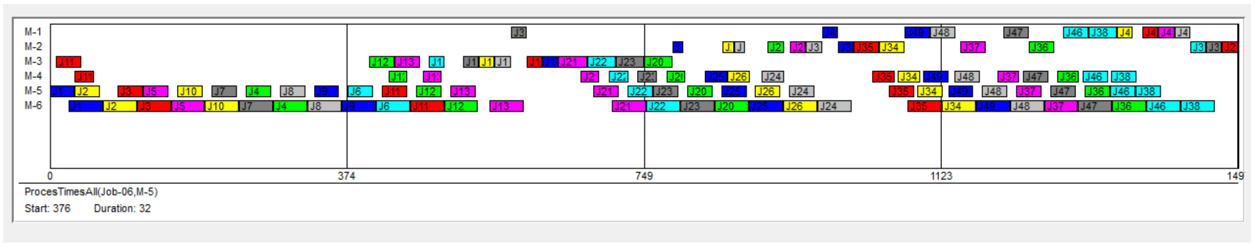
Case: 5



ProcesTimesAll(Job-16,M-1)
Start: 424 Duration: 14

Simulation Results [seconds]		
10 Sec_MIP Model		1142
MIP Model Optimization	=	1142
Model Calculation Time	=	8
Emulation Old	=	**1280**
Emulation MIP	=	**1142**

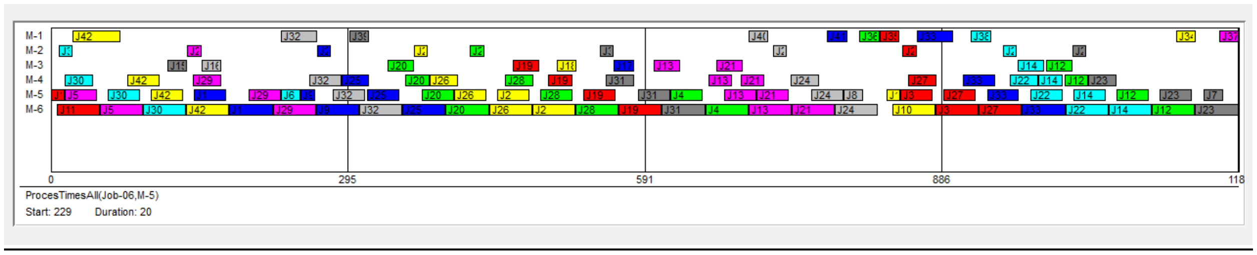
Case: 6



ProcesTimesAll(Job-06,M-5)
Start: 376 Duration: 32

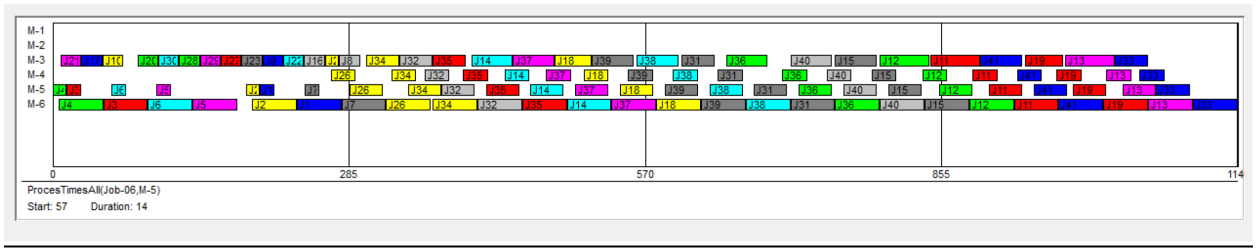
Simulation Results [seconds]		
10 Sec_MIP Model		1497
MIP Model Optimization	=	1329
Model Calculation Time	=	10
Emulation Old	=	**1812**
Emulation MIP	=	**1436**

Case: 7



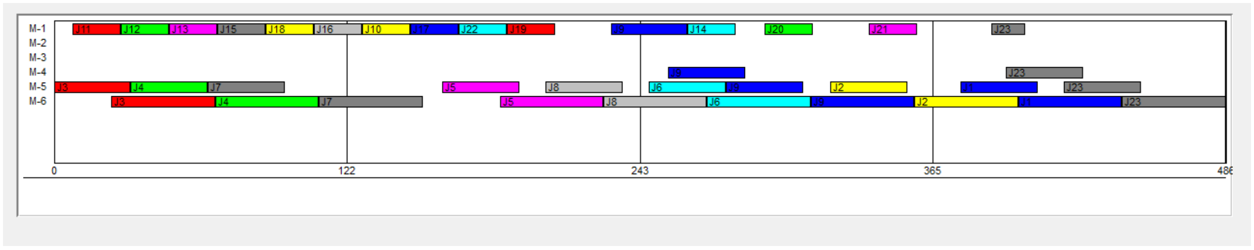
Simulation Results [seconds]		
10 Sec_MIP Model		1201
MIP Model Optimization	=	1158
Model Calculation Time	=	10
Emulation Old	=	**1476**
Emulation MIP	=	**1162**

Case: 8



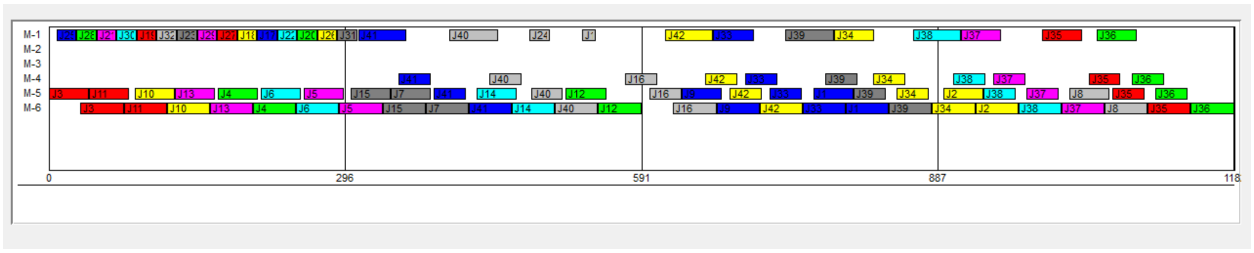
Simulation Results [seconds]		
10 Sec_MIP Model		1140
MIP Model Optimization	=	1124
Model Calculation Time	=	10
Emulation Old	=	**1368**
Emulation MIP	=	**1130**

Case: 9



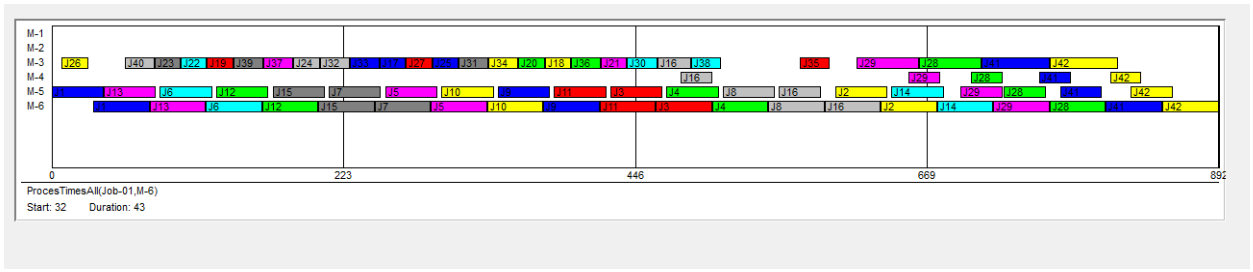
Simulation Results [seconds]		
10 Sec_MIP Model		486
MIP Model Optimization	=	454
Model Calculation Time	=	10
Emulation Old	**=**	**708**
Emulation MIP	**=**	**468**

Case: 10



Simulation Results [seconds]		
10 Sec_MIP Model		1182
MIP Model Optimization	=	1150
Model Calculation Time	=	10
Emulation Old	**=**	**1764**
Emulation MIP	**=**	**1150**

Case: 11



ProcesTimesAll(Job-01,M-6)
Start: 32 Duration: 43

Simulation Results [seconds]		
10 Sec_MIP Model		892
MIP Model Optimization	=	892
Model Calculation Time	=	10
Emulation Old	=	**1370**
Emulation MIP	=	**892**

Appendix G

```
MAIN MODEL Main_Scheduling_with_different_Machine_Groups

  DECLARATION SECTION

    SET:
       identifier   :  Jobs
       index        :  j
       order by     :  user
       definition   :  Elementrange (1,Maxjobs, prefix: "Job-")
       comment      :  "Set of all Jobs" ;

    SET:
       identifier   :  Jobs2
       subset of    :  Jobs
       index        :  k
       order by     :  user
       definition   :  Elementrange (1,Maxjobs, prefix: "Job-")  ;

    SET:
       identifier   :  JobGroups
       index        :  g
       order by     :  user
       initial data :  data { JobGroup1, JobGroup2, JobGroup3, JobGroup4, JobGroup5,
JobGroup6 } ;

    SET:
       identifier   :  JobGroups2
       subset of    :  JobGroups
       index        :  h
       order by     :  user
       definition   :  data { JobGroup1, JobGroup2, JobGroup3, JobGroup4, JobGroup5,
JobGroup6 } ;

    SET:
       identifier   :  Teller
       index        :  i
       order by     :  user
       definition   :  Elementrange (1,1000)  ;

    SET:
       identifier   :  Machines
       index        :  m
       order by     :  user
       definition   :  Elementrange (1,Maxmachines, prefix: "M-")
       comment      :  "Set of all Machines" ;

    SET:
       identifier   :  Machine1
       subset of    :  Machines
       index        :  m1
       order by     :  user ;

    SET:
       identifier   :  MachineGroups
       index        :  mg
       order by     :  user ;

    SET:
```

```
   identifier   :  PositionOfOutfeed
   index        :  p
   order by     :  user
   definition   :  Elementrange (1,MaxJobs, prefix: "Pos-")
   comment      :  "Set of all positions" ;

PARAMETER:
   identifier   :  p_time_limit
   definition   :  10 ;

PARAMETER:
   identifier   :  MaxJobs
   range        :  integer
   initial data :  100 ;
   comment      :  "Number of elements in the set Jobs" ;

PARAMETER:
   identifier   :  JobLocation
   index domain :  (j) ;

PARAMETER:
   identifier   :  MaxMachines
   range        :  integer
   initial data :  6; ;

PARAMETER:
   identifier   :  JobGroupMatrix1
   index domain :  (j,g) ;

PARAMETER:
   identifier   :  MachineGroup1
   index domain :  (mg,m)
   range        :  binary ;

PARAMETER:
   identifier   :  MachineGroup2
   index domain :  (mg,m)
   range        :  binary ;

PARAMETER:
   identifier   :  MachineGroup3
   index domain :  (mg,m)
   range        :  binary ;

PARAMETER:
   identifier   :  MachineGroup4
   index domain :  (mg,m)
   range        :  binary ;

PARAMETER:
   identifier   :  MachineGroup5
   index domain :  (mg,m)
   range        :  binary ;

PARAMETER:
   identifier   :  ProcesTimesAll
   index domain :  (j,m)
   range        :  nonnegative ;

PARAMETER:
   identifier   :  PositionProcesTimes
   index domain :  (p,m)
   definition   :  sum(j|JobPositionSchedule(j,p),ProcesTimesAll(j,m))  ;
```

```
    PARAMETER:
       identifier   :  PositionStartTimesBinary
       index domain :  (p,m)
       range        :  nonnegative
       definition   :  PositionProcesTimes(p,m) >= 1 ;

    PARAMETER:
       identifier   :  PosTimesStart
       index domain :  (p,m)
       definition   :  StartTimes(p,m) * PositionStartTimesBinary(p,m) ;

    VARIABLE:
       identifier   :  JobPositionSchedule
       index domain :  (j,p)
       range        :  binary ;

    VARIABLE:
       identifier   :  StartTimes
       index domain :  (p,m)
       range        :  nonnegative ;

    CONSTRAINT:
       identifier   :  OneJobPerPosition
       index domain :  (p)
       definition   :  sum(j,JobPositionSchedule(j,p))=1
       comment      :  "only one Job is related to every Position" ;

    CONSTRAINT:
       identifier   :  OnePositionPerJob
       index domain :  (j)
       definition   :  sum(p,JobPositionSchedule(j,p))=1
       comment      :  "only one position is related to every job" ;

    CONSTRAINT:
       identifier   :  StartTimesConstraintM1
       index domain :  (p)
       definition   :  StartTimes(p,'M-1') <= TimeSpan  ;

    CONSTRAINT:
       identifier   :  StartTimesConstraintM2
       index domain :  (p)
       definition   :  StartTimes(p,'M-2') <= TimeSpan  ;

    CONSTRAINT:
       identifier   :  StartTimesConstraintM3
       index domain :  (p)
       definition   :  StartTimes(p,'M-3') <= TimeSpan  ;

    CONSTRAINT:
       identifier   :  MachineStartTimeM1
       index domain :  (p,m) | exists ( mg|MachineGroup1(mg,m))
       definition   :  StartTimes(p,m + 3)  >=  StartTimes(p,m) + sum(j,
ProcesTimesAll(j,m)*JobpositionSchedule(j,p))-8;


                       !if ord(m) = 1 then
                       !StartTimes (p,m + 3) >= StartTimes(p,m) + sum
(j,ProcesTimesAll(j,m)*JobPositionSchedule(j,p))
                       !elseif ord(m) = 2 then
                       !StartTimes (p,m + 2) >= StartTimes(p,m) + sum
(j,ProcesTimesAll(j,m)*JobPositionSchedule(j,p))
                       !else
                       !StartTimes (p,m + 1) >= (StartTimes(p,m)  + sum
(j,ProcesTimesAll(j,m)*JobPositionSchedule(j,p)))

                       !endif
                       ;

    CONSTRAINT:
       identifier   :  MachineStartTimeM2
       index domain :  (p,m) | exists ( mg|MachineGroup2(mg,m))
       definition   :  StartTimes(p,m + 2)  >=  StartTimes(p,m) + sum(j,
ProcesTimesAll(j,m)*JobpositionSchedule(j,p))-8 ;

    CONSTRAINT:
       identifier   :  MachineStartTimeM3
       index domain :  (p,m) | exists ( mg|MachineGroup3(mg,m))
       definition   :  StartTimes(p,m + 1)  >=  StartTimes(p,m) + sum(j,
ProcesTimesAll(j,m)*JobpositionSchedule(j,p))-8 ;

    CONSTRAINT:
       identifier   :  MachineStartTimeM4
       index domain :  (p,m) | exists ( mg|MachineGroup4(mg,m))
       definition   :  StartTimes(p,m + 1)  >=  StartTimes(p,m) + sum(j,
ProcesTimesAll(j,m)*JobpositionSchedule(j,p))-8 ;

    CONSTRAINT:
       identifier   :  MachineStartTimeM5
       index domain :  (p,m) | exists ( mg|MachineGroup5(mg,m)) and m <>
last(machines)
       definition   :  StartTimes(p,m + 1)  >=  StartTimes(p,m) + sum(j,
ProcesTimesAll(j,m)*JobpositionSchedule(j,p))-8 ;

    CONSTRAINT:
       identifier   :  ScheduleStartTimeM1
       index domain :  (p,m) | p <> last(PositionOfOutfeed)
       definition   :  StartTimes(p + 1,m) >=  StartTimes(p,m) + sum(j,
ProcesTimesAll(j,m)*JobpositionSchedule(j,p)) ;

    CONSTRAINT:
       identifier   :  JobStartTimeM1
       index domain :  (p,m) | p <> last(PositionOfOutfeed) and exists
(mg|MachineGroup1(mg,m))
       definition   :  if PositionProcesTimes(p,m) >=0 and exists
(mg|MachineGroup1(mg,m)) then

                       StartTimes(p+1,'M-1') >= StartTimes(p,'M-4')+8
                       else
                       StartTimes(p+1,m) >=0
                       endif ;

    CONSTRAINT:
       identifier   :  JobStartTimeM2
       index domain :  (p,m) | p <> last(PositionOfOutfeed) and exists
(mg|MachineGroup2(mg,m))
       definition   :  if PositionProcesTimes(p,m) >=0 and exists
(mg|MachineGroup2(mg,m)) then

                       StartTimes(p+1,'M-2') >= StartTimes(p,'M-4')+8
                       else
                       StartTimes(p+1,m) >=0
                       endif ;

    CONSTRAINT:
       identifier   :  JobStartTimeM3
       index domain :  (p,m) | p <> last(PositionOfOutfeed) and exists
(mg|MachineGroup3(mg,m))
       definition   :  if PositionProcesTimes(p,m) >=0 and exists
(mg|MachineGroup3(mg,m)) then

                       StartTimes(p+1,'M-3') >= StartTimes(p,'M-4')+8
                       else
                       StartTimes(p+1,m) >=0
                       endif ;

    CONSTRAINT:
       identifier   :  JobStartTimeM4
       index domain :  (p,m) | exists (mg|MachineGroup4(mg,m)) and p <>
last(PositionOfOutfeed)
       definition   :  if PositionProcesTimes(p,m) >=1 and exists
(mg|MachineGroup4(mg,m)) then

                       StartTimes(p+1,'M-4') >= StartTimes(p,'M-5')+8
                       else
                       StartTimes(p+1,m) >=0
                       endif ;

    CONSTRAINT:
       identifier   :  JobstartTimeM5
       index domain :  (p,m) | exists (mg|MachineGroup5(mg,m)) and p <>
last(PositionOfOutfeed)
       definition   :  if PositionProcesTimes(p,m) >=1 and exists
(mg|MachineGroup5(mg,m)) then

                       StartTimes(p+1,m) >= StartTimes(p,m +1)+8
                       else
                       StartTimes(p+1,m) >=0
                       endif ;

    CONSTRAINT:
       identifier   :  DigOut1
       index domain :  (j,p,k,g,h)|(ord(g)<card(g)) and (ord(h)>ord(g)) and
(ord(j)<>ord(k))!|JobGroup1(j)=1
       definition   :  if JobGroupMatrix1(j,g)=1 and JobGroupMatrix1(k,h)=1  and
(JobLocation(j) = JobLocation(k)) then
                       JobGroupmatrix (k) >= JobPositionSchedule(j,p)* ord (p)

                       elseif
                       JobGroupMatrix1(j,g)=2 and JobGroupMatrix1(k,h)=2  and
(JobLocation(j) = JobLocation(k)) then
                       JobGroupmatrix (k) >= JobPositionSchedule(j,p)* ord (p)

                       elseif
                       JobGroupMatrix1(j,g)=3 and JobGroupMatrix1(k,h)=3  and
(JobLocation(j) = JobLocation(k)) then
                       JobGroupmatrix (k) >= JobPositionSchedule(j,p)* ord (p)

                       elseif
                       JobGroupMatrix1(j,g)=5 and JobGroupMatrix1(k,h)=5  and
(JobLocation(j) = JobLocation(k)) then
                       JobGroupmatrix (k) >= JobPositionSchedule(j,p)* ord (p)

                       else
                       JobGroupmatrix (k) >= 0
                       !JobPositionSchedule(j,p) * ord (p) >= 0
                       endif ;

    VARIABLE:
       identifier   :  TimeSpan
       range        :  free
       definition   :  StartTimes(last(p),last(m)) + sum(j,
ProcesTimesAll(j,last(m))*JobPositionschedule(j,last(p))) !+
sum((p,m),StartTimes(p,m))
       comment      :  "Total time to process all jobs on the machines (we are going
to minimize this)" ;

    MATHEMATICAL PROGRAM:
       identifier   :  CalculateStartTimes
       objective    :  TimeSpan
       direction    :  minimize
       constraints  :  AllConstraints
       variables    :  AllVariables
       type         :  Automatic ;

    VARIABLE:
       identifier   :  JobGroupMatrix
       index domain :  (j)
       range        :  free
       definition   :  sum (p,JobPositionSchedule(j,p) * ord (p)) ;

  ENDSECTION  ;

  DECLARATION SECTION Interface_Declaration

    PARAMETER:
       identifier   :  InterfaceStartTime
       index domain :  (j,m)
       definition   :  StartTimes(first(p|JobPositionSchedule(j,p)),m) ;

    PARAMETER:
       identifier   :  InterfaceProcessTime
       index domain :  (p,m)
       definition   :  sum(j|JobPositionSchedule(j,p),ProcesTimesAll(j,m))  ;

    STRING PARAMETER:
       identifier   :  JobDescriptionForGantChart
       index domain :  (j)
       definition   :  FormatString("J%i", ord(j))
       comment      :  "Form the Job-name from Job-1 to J1" ;

    STRING PARAMETER:
       identifier   :  MachineDescrptionForGantChart
       index domain :  (m)
       definition   :  FormatString("M%i", ord(m))
       comment      :  "Form the Machine-name from M-1 to M1" ;

  ENDSECTION  ;

  SECTION Excel_Linkage


    DECLARATION SECTION

      STRING PARAMETER:
         identifier   :  WorkbookName
         initial data :  "InlezenVanafExcel.xls" ;

    ENDSECTION  ;

    PROCEDURE
      identifier :  ReadDataFromExcel
      body       :
        Spreadsheet::SetActiveSheet(WorkbookName, "ProcesTimesAll");

        Spreadsheet::RetrieveParameter(WorkbookName, ProcesTimesAll, "C4:H23");
        Spreadsheet::RetrieveParameter(WorkbookName, JobLocation, "C4:C23",
"JobLocation");

        Spreadsheet::RetrieveParameter(WorkbookName, JobGroupMatrix1, "C5:H24",
"JobGroupMatrix1");


        Spreadsheet::CloseWorkbook(WorkbookName,0);

    ENDPROCEDURE  ;

    PROCEDURE
      identifier :  CloseWorkbook
      body       :
        Spreadsheet::CloseWorkbook(WorkbookName,0);

    ENDPROCEDURE  ;

  ENDSECTION Excel_Linkage ;

  PROCEDURE
    identifier :  MainInitialization

  ENDPROCEDURE  ;

  PROCEDURE
    identifier :  MainExecution
    body       :
      solve CalculateStartTimes where !MIP_Relative_Optimality_Tolerance := 0.05;
      !time_limit := p_time_limit;
      time_limit := 1000 ;

  ENDPROCEDURE  ;

  PROCEDURE
    identifier :  MainTermination
    body       :
      return DataManagementExit();

  ENDPROCEDURE  ;

ENDMODEL Main_Scheduling_with_different_Machine_Groups ;
```