Study on the Design and Simulation of a High capacity transhipment terminal

- Report

Impression of the portainers and container warehouse

December 2000
Hein van Hees
Thies Visscher

TU Delft
Deel University of Technology

Faculty of Civil Engineering and Geosciences
Department of Hydraulic and Geotechnical Engineering
Section of Hydraulic Engineering
Foreword

This report concerns the study on the design and simulation of a high capacity container terminal. It has been carried out as a Master of Science thesis for section Ports and Terminals of the faculty of Civil Engineering and Geosciences at the Delft University of Technology. The appendices of this report have been divided into two separate volumes.

We would like to thank prof. ir. H. Ligtering and ir. R. Groenveld of the section Ports and Waterways and prof. ir. J.C. Rijsebrij of the section Transport technology of the faculty of Mechanical Engineering and Marine Technology for their support and critical notes. We would like to thank prof. Hoong Chor Chin for the efforts he made as this study has been partly carried out at the National University of Singapore.

Finally we would like to thank our parents for their support, and especially Willem Visscher for his enthusiastically given advises.

Hein van Hees
Thies Visscher
Delft, December 2000
Contents

Foreword

Summary

List of Figures

List of tables

List of Definitions

1 Introduction ................................................................................................................. 1
2 Problem analysis ......................................................................................................... 5
   2.1 Objective ............................................................................................................. 5
   2.2 Problem approach ............................................................................................... 5
3 Assumptions and boundary conditions concerning the terminal ............................... 7
   3.1 Containers ......................................................................................................... 7
   3.2 The vessels ......................................................................................................... 8
      3.2.1 The Mega vessel ........................................................................................... 8
      3.2.2 The feeders .................................................................................................. 9
   3.3 The berths ........................................................................................................... 9
   3.4 Handling strategies of the portainers .................................................................... 10
   3.5 Terminal strategies .............................................................................................. 11
   3.6 Arrival cycles of the vessels ................................................................................ 15
   3.7 Required container storage capacity ...................................................................... 16
4 Choice of the terminal components and their capacities ........................................... 17
   4.1 Portainers .......................................................................................................... 17
   4.2 Container storage system .................................................................................... 21
      4.2.1 Description of the chosen warehousing variant ............................................. 22
      4.2.2 Adaptations to the chosen warehousing concept .......................................... 25
   4.3 Horizontal transportation ...................................................................................... 28
      4.3.1 Rubber tired automated guided vehicles (agv's) ............................................ 29
      4.3.2 Rail guided vehicles (rgv's) ........................................................................... 30
      4.3.3 Criteria and choice ....................................................................................... 31
5 The design of the terminal Layout ............................................................................ 33
   5.1 Positioning of the container warehouses ............................................................ 33
   5.2 Positioning and capacity of the rgv lanes ............................................................ 34
      5.2.1 Inter-berth lanes ............................................................................................ 34
      5.2.2 Yard-lanes ..................................................................................................... 36
   5.3 Additional terminal design aspects ....................................................................... 38
   5.4 Handling strategies applied to the terminal Layout .............................................. 40
      5.4.1 The unloading of a container vessel ............................................................... 40
      5.4.2 The adjacent loading of a mega vessel ........................................................ 41
      5.4.3 The adjacent loading of a feeder vessel ......................................................... 42
      5.4.4 The shuffle process on the terminal ............................................................... 42
   5.5 Number of warehouse cranes .............................................................................. 43
   5.6 Number of rgv's on the terminal ........................................................................ 45
6 Computer simulation ................................................................................................. 47
   6.1 Problem analysis tool: PROSIM ......................................................................... 47
   6.2 Definition of the components of the computer model ............................................ 48
      6.2.1 The component Main .................................................................................... 49
      6.2.2 The component ship Generator .................................................................... 50
      6.2.3 Import generator & Export generator ............................................................ 50
      6.2.4 The component Berthmaster ....................................................................... 51
      6.2.5 The component Portainermaster ................................................................. 52
      6.2.6 The component Rgymaster ......................................................................... 56
      6.2.7 The component Trafficmaster .................................................................... 58
      6.2.8 The class component Vessel ....................................................................... 61
      6.2.9 The class component Portainer .................................................................... 62
Foreword

This report concerns the study on the design and simulation of a high capacity container terminal. It has been carried out as a Master of Science thesis for section Ports and Terminals of the faculty of Civil Engineering and Geosciences at the Delft University of Technology. The appendices of this report have been divided into two separate volumes.

We would like to thank prof. ir. H. Lichteringen and ir. Groenveld of the section Ports and Terminals and prof. ir. J.C. Rijsenbrij of the section Transport technology of the faculty of Mechanical Engineering and Marine Technology for their support and critical notes. We would like to thank prof. Hoong Chor Chin for the efforts he made as this study has been partly carried out at the National University of Singapore.

Hein van Hees
Thies Visscher
Delft, December 2000
Summary

The container traffic growth worldwide continues to increase. To keep up with the growth, shipping companies keep enlarging container ship size. The handling speed of the terminal will especially be a critical factor for the largest vessels. Terminals have to be able to serve these ships as best as possible. To keep the dwell time of the ships visiting a terminal to a minimum, the handling capacities on the terminal must get bigger. To achieve a maximum effectiveness of portainer productivity in relation to the land side of the terminal, the horizontal transportation and the stacking of the containers must be optimised. The amount of land available for expanding an existing terminal or creating a new terminal is scarce. When designing a terminal this is a very important factor to take into account. To minimise operating costs and maximise efficiency, automation of the terminal also is a factor to take into account.

The goal of this study has been to design a container transhipment terminal capable of handling mega container vessels and analysing the logistic processes that take place on this terminal. The analysis will be performed by a computer simulation in the simulation language PROSIM.

The designed transhipment terminal consists of three berths, two feeder berths and one mega vessel berth. The ship to shore container handling takes place by using portainers. The horizontal transportation takes place by rail mounted automated guided vehicles (rgv's) that use linear motor based transfer technology as their source of propulsion. The rgv's take corners by having their wheels rotated by four disks integrated in the rail system. Container storage takes place by stacking containers in racks placed in warehouses. This allows random access to the containers eliminating the time consuming shuffling.

With these components a terminal layout has been designed. Based on this layout a computer simulation has been carried out in order to analyse the logistic processes on the terminal.

Having assumed an average portainer capacity of 1 move per minute the achieved handling rates lie between 194 moves per hour and the 236 moves per hour for the feeder vessel berths, where the handling takes place by four portainers.

The handling rates on the mega vessel berth lie between the 300 moves per hour and the 352 moves per hour. Here the vessel handling takes place by 6 portainers. Portainer occupancy rates of 95-98% have been achieved during the unloading process of the container vessels.
List of Figures

Figure 1.1, development of container vessels ............................................................................. 1
Figure 1.2, hub and spoke system .................................................................................................. 2
Figure 1.3, map of Singapore and its region South-east Asia ....................................................... 3
Figure 3.1, mega vessel .................................................................................................................. 8
Figure 3.2, tangent berth [1], perpendicular berth [2] and a combination [3] ............................... 10
Figure 3.3, schematisation of transhipment terminal ................................................................. 11
Figure 3.4, a three berth terminal ................................................................................................ 12
Figure 3.5, the fast unloading strategy/adjacent loading strategy ............................................... 13
Figure 3.6, adjacent unloading strategy and fast loading strategy ........................................... 13
Figure 3.7, the fast loading, fast unloading and shuffle strategies ........................................... 14
Figure 4.1, Dual hoist system with a fixed platform ..................................................................... 17
Figure 4.2, dual hoist system with elevating platform ................................................................ 18
Figure 4.3, double trolley system .................................................................................................. 18
Figure 4.4, elevating girder crane ............................................................................................... 19
Figure 4.5, Paceco supertainer .................................................................................................... 19
Figure 4.6, the portainer cycle times ........................................................................................... 20
Figure 4.7, cross-section of a warehouse ...................................................................................... 22
Figure 4.8, top view of a warehouse ............................................................................................ 22
Figure 4.9, the storage rack of a warehouse ................................................................................ 23
Figure 4.10, front view of the lifting process of a shuttle ............................................................ 24
Figure 4.11, the cradle of the elevating transfer vehicle .............................................................. 24
Figure 4.12, a cross section of two warehouses ......................................................................... 26
Figure 4.13, top view of the warehouse with adaptations ........................................................... 26
Figure 4.14, agv turning radius .................................................................................................... 29
Figure 4.15, rgy taking a turn ....................................................................................................... 30
Figure 5.1, first design phase ......................................................................................................... 34
Figure 5.2, schematisation of inter-berth rgy flow when three vessels are being adjacent loaded ................................................................................................................................. 35
Figure 5.3, Layout of Berth 2 including inter-berth lanes ............................................................ 35
Figure 5.4, Layout version 1 including warehouse lanes .............................................................. 36
Figure 5.5, layout version 2 including warehouse lanes ............................................................... 37
Figure 5.6, side view of the portainer .......................................................................................... 38
Figure 5.7, Berth 1 ........................................................................................................................ 39
Figure 5.8, unloading strategy of a container vessel .................................................................... 40
Figure 5.9, adjacent loading from Yard 1 to Berth 2 ................................................................. 41
Figure 5.10, adjacent loading of a feeder vessel from Yard 2 .................................................... 42
Figure 5.11, portainer production during unloading/fast-loading ................................................ 43
Figure 5.12, Portainer production during unloading fast loading on Berth 1 and adjacent loading on Berth 2 .................................................................................................................... 44
Figure 5.13, Portainer production during unloading/fast-loading on berth 2 and adjacent loading on berth 1 and berth 3 ......................................................................................................... 44
Figure 6.1, the process of Main .................................................................................................... 49
Figure 6.2, process of the ship Generator ................................................................................... 50
Figure 6.3, Process of the Berthmaster ...................................................................................... 51
Figure 6.4, mega vessel moored at Berth 2 ............................................................................... 52
Figure 6.5, feeder vessel moored at Berth 1 .............................................................................. 52
Figure 6.6, a feeder moored at Berth 3 ...................................................................................... 53
Figure 6.7, feeder vessels moored at Berth 1 & Berth 3 ............................................................. 53
Figure 6.8, feeder vessel moored at Berth 1 and a mega vessel moored at Berth 2 ............... 54
Figure 6.9, feeder vessel moored at Berth 3 and a mega vessel moored at Berth 2 ............... 54
Figure 6.10, Three vessels moored ............................................................................................ 54
Figure 6.11, process of the Rgymaster ....................................................................................... 57
Figure 6.12, process of the Trafficmaster .................................................................................. 58
Figure 6.13, possibilities when a rgy is working for a vessel ..................................................... 59
Figure 6.14, process of the class component vessel ................................................................. 61
Figure 6.15, process of the class component portainer .................................................. 62
Figure 6.16, flow chart of the portainer while it is going to the right position at the quay ...... 63
Figure 6.17, process of the rgv ...................................................................................... 65
Figure 6.18, process of the class component crane ......................................................... 66
Figure 6.19, the creation of a main track by the Main ..................................................... 68
Figure 6.20, input file for creating warehouse lanes ...................................................... 69
Figure 6.21, the creation of a stack ................................................................................. 69
Figure 6.22, input file for creating shortcuts .................................................................. 70
Figure 6.23, the creation of a shortcut ........................................................................... 70
Figure 6.24, input file for creating a platform ............................................................... 71
Figure 6.25, the creation of 2 platforms and a portainer location .................................... 71
Figure 6.26, the claiming of nodes when driving on the main track.............................. 72
Figure 6.27, space for waiting queue at node 44 ............................................................ 73
Figure 6.28, flow chart of the driving process of a rgv ................................................. 74
Figure 6.29, the driving of the rgv ................................................................................ 75
Figure 7.1, allocation of stacks in Yard 2 during the unloading of a mega vessel ............. 79
Figure 7.2, allocation of the stacks during the adjacent loading of a feeder vessel moored at Berth 1 ............................................................................................................ 80
Figure 7.3, rgv entries in the stacks during a shuffle process ......................................... 81
Figure 7.4, the queue behind a claimed node ................................................................. 83
Figure 7.5, unloading cycle of a rgv .............................................................................. 84
Figure 7.6, rgv unloading cycle times when unloading a vessel .................................... 85
Figure 7.7, loading cycle of an rgv ............................................................................... 85
Figure 7.8, rgv cycle times when adjacent loading a vessel ........................................... 86
Figure 7.9, shuffling cycle of a rgv ............................................................................... 87
Figure 7.10, rgv cycle times during shuffling ............................................................... 87
Figure 7.11, the portainer cycle times ........................................................................... 88
Figure 7.12 portainer productivity during the unloading of a feeder vessel .................. 89
Figure 7.13, portainer productivity when adjacent loading using a different number of rgv's .......................................................... 90
Figure 7.14, Portainer 1 queue using 15 rgv's per portainer during the adjacent loading of a feeder .......................................................... 90
Figure 8.1, a feeder vessel moored at Berth 1 ............................................................... 94
Figure 8.2, the unloading of a feeder vessel ................................................................. 95
Figure 8.3, The portainer occupancy rates during the unloading of a feeder vessel at Berth 1 ........................................................................................................ 95
Figure 8.4, rgv queue under the portainers at Berth 1 during the unloading of a feeder ..... 96
Figure 8.5, rgv waiting queues on warehouse lane 22 ................................................... 96
Figure 8.6, the fast loading of a feeder vessel ............................................................... 97
Figure 8.7, the portainer occupancy rates during the fast loading of a feeder vessel ....... 97
Figure 8.8, rgv waiting rows under portainers at Berth 1 when loading fast ............... 98
Figure 8.9, the adjacent loading of a Feeder .................................................................. 99
Figure 8.10, the portainer occupancy rates when loading a feeder from an adjacent berth . 99
Figure 8.11, the rgv waiting queues under the portainers at Berth 1 when adjacent loading 100
Figure 8.12, the shuffling from Yard 1 ........................................................................... 101
Figure 8.13, rgv queue at node 290 .............................................................................. 102
Figure 8.14, rgv waiting row on warehouse lane 21 ...................................................... 102
Figure 8.15, mega vessel moored at Berth 2 .................................................................. 105
Figure 8.16, the unloading of a mega vessel ................................................................ 106
Figure 8.17, the occupancy rate of portainers 5 to 10 ................................................... 106
Figure 8.18, rgv waiting queues under the portainers on Berth 2 .............................. 107
Figure 8.19, the fast loading of a mega vessel .............................................................. 108
Figure 8.20, Portainer occupancy rates during the fast loading of a mega vessel .......... 108
Figure 8.21, the rgv waiting queues under the portainers when loading a mega vessel .... 109
Figure 8.22, the adjacent loading of a mega vessel ..................................................... 110
Figure 8.23, Portainer occupancy rate when loading a mega vessel from adjacent berths . 110
Figure 8.24, the rgv waiting queues under the portainers during the loading from an adjacent berth .......................................................... 111
Figure 8.25, shuffling from Berth 2 ............................................................................. 112
Figure 8.26, the rgv queue at node 363 when shuffling from Berth 2 ......................... 112
Figure 8.27, the rgv queue at node 393 during shuffling from Berth 2 ......................... 113
List of Tables

Table 3.1, Results of cycle times calculations .......................................................... 15
Table 4.1, operational properties of a rgv ................................................................. 29
Table 5.1, Warehouse crane occupancy rates ............................................................ 44
Table 5.2, number of needed rgv's per portainer ......................................................... 45
Table 8.1, the undisturbed container productions for Berth 1 .................................. 103
Table 8.2, warehouse crane occupation rates during the undisturbed handling of a feeder vessel .................................................................................. 104
Table 8.3, berth productions for Berth 2 ................................................................... 114
Table 8.4, warehouse crane occupation rates during the undisturbed handling of a mega vessel .................................................................................. 115
Table 8.5, Experiments with 3 occupied berths ......................................................... 116
Table 8.6, berth productions during adjacent loading on all berths ............................. 118
Table 8.7, berth productions ...................................................................................... 122
Table 8.8, berth productions ...................................................................................... 126
Table 8.9, handling rates during simulation on 3 berths ........................................... 130
Table 8.10, comparison between the undisturbed and maximum service times .......... 130
Table 8.11, warehouse crane occupancy rates ........................................................... 131
Table 8.12, description of alterations to the reference terminal ................................. 134
Table 8.13, handling productions with 12 rgv's .......................................................... 135
Table 8.14, the mean portainer cycle times ................................................................. 139
Table 8.15, portainer productions during the basic run and the crane experiment ........ 143
Table 9.1, the handling rates and service times ......................................................... 146
Table 9.2, comparison between undisturbed situation and the ideal preliminary calculation .............................................................. 147
Table 9.3, comparison between undisturbed situation and the ideal preliminary calculation .............................................................. 147
Table 9.4, comparison between disturbed situation and the ideal preliminary calculation .............................................................. 148
Table 9.5, comparison between disturbed situation and the ideal preliminary calculation .............................................................. 148
Table 9.6, portainer productions of the basic run and the crane experiment .............. 150
Table 9.7, number of cranes and their occupancy rates .............................................. 150
Table 9.8, simulation results for the waiting rows and applied waiting row length .......... 151
### List of Definitions

<table>
<thead>
<tr>
<th>Definition</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Berth:</strong></td>
<td>Location where the vessels will moor.</td>
</tr>
<tr>
<td><strong>Yard:</strong></td>
<td>Behind each berth there is a yard containing the container storage.</td>
</tr>
<tr>
<td><strong>Stack:</strong></td>
<td>The yards are divided into two stacks; A stack that holds the 20’ containers and a stack that holds the 40’ containers.</td>
</tr>
<tr>
<td><strong>Portainer lane:</strong></td>
<td>Rgv are handled by a portainer on the portainer lanes located behind the portainers.</td>
</tr>
<tr>
<td><strong>Yard lane:</strong></td>
<td>A yard lane is a perpendicular to the berth orientated lane that allows the rgv’s to enter the yard.</td>
</tr>
<tr>
<td><strong>Inter-berth lane:</strong></td>
<td>An inter-berth lane provides the connection between the 3 yard-berth combinations.</td>
</tr>
<tr>
<td><strong>Warehouse lane:</strong></td>
<td>A warehouse lane allows the rgv’s to enter the container warehouses from a yard lane.</td>
</tr>
<tr>
<td><strong>Reference terminal:</strong></td>
<td>The terminal defined as standard terminal. The reference terminal is the basis of each simulation run.</td>
</tr>
</tbody>
</table>
1 Introduction

The transportation of goods in containers has revolutionised the maritime and port business across the world. Since the introduction of container transportation in the sixties, the development of container transport has increased dramatically. Nowadays container transportation is a full-grown business with a high level of technological means.

This increase of container transportation and subsequently the growing capacity of container vessels over the last couple of years has forced terminal operators to constantly search for means to increase their handling capacity. Already 8,000 and 8,000+ TEU\(^1\) mega vessels are on the drawing boards. Add to this the ever-growing number of 6,000 and 6,600 TEU ships already in operation and it becomes clear that shipping operators are increasing their ship sizes at a fast rate, as illustrated in Figure 1.1.

![Development of container vessels](image)

Figure 1.1, development of container vessels

The size of these ships and the enormous costs per hour, demand an optimal turn around time. Due to the increase in scale, terminals with high loading and unloading capacities will have the advantage in competition. Especially those terminal operators wishing to maintain their ‘hub port’ status should upgrade their facilities, as shipping companies will prefer to attend terminals where vessel handling times are short. The hub ports offering the fastest, most reliable service at lowest overall cost will attract the biggest share of container traffic.

Overpopulation and industrial development mean that the amount of land available for the construction or expansion of container terminals is minimal, whilst environmental concerns place new restrictions on development operations. Particularly in Singapore, Hong Kong and perhaps in lesser degree also Rotterdam port area is a scarce commodity. Massive land reclaiming projects are needed for port expansion. It is therefore essential that along with increased handling speed the land usage of the terminal needs to be minimised in the terminal design.

Besides the need for higher handling speed and optimisation of land usage a third trend in port development can be distinguished. In their attempts to control the complete range of terminal operations, ports see their efforts frustrated by the variability of man operated vehicles. Automation will eliminate this and should facilitate the development of efficient

\(^1\) TEU = Twenty foot Equivalent Unit
scheduling of operating systems. Automation of terminal operations also reduces operating costs by removing manpower besides satisfying the growing demand for efficiency.

The expectation is that in a number of years, mega vessels with capacities of 8,000 TEU or more will maintain the intercontinental container transportation between ports with a hub status. Large amounts of containers will have to be handled at each mega vessel call. Feeder vessels responsible for the regional distribution of containers will serve the regional ports, as is schematised in Figure 1.2.

![Figure 1.2, hub and spoke system](image)

A typical hub port is the transhipment terminal. A transhipment terminal has no or limited hinterland demanding containers. The container flows to and from the terminal are limited to the unloading and the loading of the container vessels. Good examples of transhipment hub ports are the container terminals of Singapore. Goods manufactured in Southeast Asia are brought to Singapore from regional ports by feeder vessels. From Singapore the loaded containers are distributed all over the world. In this study the situation as it occurs in Singapore will be used as a basis. Figure 1.3 demonstrates the central location of Singapore within Southeast Asia.
This study contains the following chapters. The problem analysis will be dealt with in Chapter 2, followed by a description of the specifications on and around the terminal in Chapter 3. Chapter 4 describes the components of container, container storage system, and horizontal transportation. The chosen components will be integrated into a terminal layout in Chapter 5. Chapter 6 describes the computer simulation model, which will be verified in Chapter 7. The standard reference terminal is defined in the first part of Chapter 8 and will be analyzed by means of computer simulation in the second part of Chapter 8. Discussion of the results is carried out in Chapter 9, followed by the conclusions in Chapter 10 and the recommendations in Chapter 11.
2 Problem analysis

2.1 Objective

The objective of this study is to design and simulate a transhipment container terminal:

- Capable of handling mega container vessels of 8000 TEU.
- With a high level of automation.
- While maximising container throughput.
- While minimising the land usage.
- With the intention not to invent new kinds of terminal equipment. The aim is to use only existing technology.

The attention will be especially focused on the container flows at the terminal and the achievements of the different components in relation to each other.

2.2 Problem approach

The container terminal will have comparable circumstances as to those in Singapore. This means that any transportation of containers to hinterland can be neglected and will not be taken into account, making it a pure transhipment terminal called at by regional feeders and intercontinental container vessels.

The first phase will be the design process of the transhipment container terminal by first choosing the terminal components followed by integrating these components into a terminal layout. After the design of the terminal is complete, the operational processes of the terminal will be modelled and simulated with a computer simulation program.

The second phase in the design of the terminal will be to simulate the processes of the terminal with the computer simulation program PROSIM. This way the terminal capacities can be analysed and quantified.

This study will be highly concentrated on the processes that take place in the terminal. This means that a high level of simulation accuracy will be achieved concerning the different components at the terminal. A result of this design approach is that in the simulation phase, the terminal will be primarily tested by temporarily increasing and decreasing the pressure on the terminal and not by performing simulation runs that comprises several weeks of arriving container vessels.

This makes it more difficult to make comments on vessel turn around times, stack capacities and yearly throughput, as longer runs are required. But it does give the possibility to look at the specific processes at the terminal in detail.
3 Assumptions and boundary conditions concerning the terminal

During the design process of a terminal some boundary conditions must be understood and assumptions must be made. These boundary conditions concern the type of vessels that call on the terminal and the different kinds of containers they deliver. The assumptions concern some choices with respect to the number of berths and the handling strategies at the terminal.

3.1 Containers

Currently a number of different common container types can be distinguished by size and sort of load. An inventory has been made by [L.1].

- Standard containers.
  Containers that belong to this group have a length of 20’ or 40’ and a width of 8’. The height varies between 8’ and 9’6”. Electrically cooled containers or reefers also belong to this group, and need special facilities to provide them with electricity. 98% of all containers belong to this category. The amount of 20’ containers handled on the terminal is the same as the amount of 40’ containers handled on the terminal.

- Containers with a 45’ length.
  Except for the length these containers do not differ from the standard containers. Of all handled containers a mere 1% belongs to this category.

- Off standard A containers.
  Containers belong to this class if the width exceeds the standard width with half a meter. If the length of a container exceeds a 40’ standard container with one meter or if the length exceeds a 45’ container by half a meter it also belongs to this category. If the standard height is exceeded by a meter, if the height exceeds 3.6 meters or if the container weighs more than 41.5 tons the container also belongs to this category. 0.8% of all containers belongs this category.

- Off standard B containers.
  If the dimensions of the container do not correspond with any of the above classes, it automatically belongs to this category. 0.2% of these containers belong to this group of off-standard containers.

- IMO containers.
  Containers containing hazardous loads are classified as IMO containers. Containers belonging to IMO class 1 and 7 require a special position on the container vessel. The dimensions of these containers however almost always correspond to standard containers. 0.1% of the containers belong to this category.

Both the off standard B and the certain IMO classes of containers require off-standard transportation facilities. The off standard exceptions will not be included in the simulation program. The simulation will only include the standard 20’ and 40’ containers. 45’ containers will be treated the same way as 40’ containers.
3.2 **The vessels**

3.2.1 **The Mega vessel**

As demand for container transportation increased over the years the vessel size followed. Currently the feasibility of mega vessels with capacities varying from 8,000 TEU up to 10,000 TEU is being studied on, while 6,700 TEU vessels are already operational. In this study we will derive the design vessel from the FAMAS project, being an 8,000 TEU mega container vessel. The mega vessel as used in FAMAS is based on a number of assumptions stated in [L.2], [L.3] and [L.4].

- The length of the mega vessels will not exceed 350 meters due to shipping characteristics.
- The depth of the mega vessels will be restricted to 14 meters considering the water depth of most harbours.
- Inside the hold of the mega vessels there is room for stacking 9 containers in height.
- On the decks the maximum stacking height is 7 containers.
- There are six bays behind the wheelhouse allowing two portainers to handle the rear of the vessel.

Based on these assumptions the following vessel properties have been found.

<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Capacity</td>
<td>8,000 TEU</td>
</tr>
<tr>
<td>Length:</td>
<td>335 m.</td>
</tr>
<tr>
<td>Width:</td>
<td>53 m.</td>
</tr>
<tr>
<td>Draught:</td>
<td>14 m.</td>
</tr>
</tbody>
</table>

![Image of the mega vessel](image)

_Figure 3.1, mega vessel_

These vessels will be responsible for the intercontinental hub to hub transportation of the containers.
3.2.2 The feeders

The feeders take care of the regional distribution of containers and attend regional ports. Some assumptions concerning the feeders are made.

- The feeders are 2nd or 3rd generation container vessels. These vessels can carry around 1600 TEU
- The length of the feeders varies between 201 and 230 meters.
- The depth of the feeders will not exceed 11.5 meters
- The width of the feeders will not exceed 32 meters.

This study has been based on the situation as it exists in Singapore, so the share of containers of the terminal on the mega vessels will correspond to the current share of containers on the intercontinental container trade. This share is roughly 60 percent resulting in 4800 TEU to be unloaded and 4800 TEU to be re-loaded.

A feeder vessel is supposed to only serve this transshipment terminal and will therefore unload its entire cargo at the terminal. The feeder will then be completely loaded again with empty containers.

3.3 The berths

The number of berths

Generally a container terminal and especially hub terminals have multiple berths. By combining the portainers, container storage and horizontal transportation, the larger terminals will be more efficient and flexible. To include these synergy advantages into the initial design and the computer simulation, the choice has been made to design a multiple berth terminal. Increasing the number of berths consequently will also increase the complexity and the magnitude of the simulation program. A compromise between the complexity of the simulation program and the inclusion of the synergy advances has been found by designing a three-berth terminal.

The orientation of the berths

Orientation of berths can roughly be divided into the conventional tangent orientation and perpendicular orientation or slip berth. Lately a lot of research has been done to the nautical, hydrodynamic and logistic aspects of slip berths. A big advantage is that a vessel can be handled from both sides. Unfortunately, docking a ship between two quays in tidal water will not be an easy matter to arrange. Other disadvantages are:

- The berth occupancy and portainer occupancy is much lower as portainers cannot be transferred from one berth to another.
- The berth is not suitable for all vessel sizes.
- Twice as much quay walls are required.

As it is still uncertain if the advantages outweigh the disadvantages of the berths, the design and simulation of the terminal will be carried out with conventional tangent orientated berths as can be seen in Figure 3.2-1.
The berth length

Normally, the berth length depends on the length of the design vessel, which can be determined by extensive research on the arrival frequencies and the length of the visiting container vessels. As the type of the container vessels in this study are schematised and restricted, and arrival times will be left out of the computer simulation, a realistic and practical design vessel cannot be determined this way.

In practise only one mega vessel can be berthed at a time. The transhipment character of the terminal justifies to assume that the other two available berths are taken by feeders.

The complete berth length thus equals the length of a mega vessel plus the length of two feeder vessels multiplied with a margin of 10 percent needed for mooring. This amounts to a total length of 875 meters for the 3 berths.

3.4 Handling strategies of the portainers

The unloading and loading of a container vessel is determined by the handling plan. Different strategies can be applied on a handling plan. These strategies are divided in bay level strategies and ship level strategies.

Bay level strategies.

When working at bay level there are a few strategies to be distinguished. These strategies often depend on the shape and type of a container vessel and the distribution of the containers on the vessel.

- Working vertically in the vessel. This is only possible with a hatchcoverless vessel and when unloading the hold of a conventional vessel.
- Working in the width of the vessel, the grazing of the ship. The containers are unloaded and loaded layer by layer from one side of the ship to the other side of the ship.
Other than the above bay level strategies exist, but are difficult to carry out. The chosen bay level strategy used to handle the containers on the vessels in this study is the grazing of the ship.

**Strategies at ship level**

Three possibilities can be distinguished for the handling of the containers at ship level.

- **Separate loading and unloading.**
  A portainer starts unloading a bay. If the bay is unloaded the portainer will move to a next bay and resumes the unloading process. All the portainers in this process move in the same direction. If one portainer is finished unloading its hold the loading of the holds starts. The aim for all the portainers is to start the loading process simultaneously. The portainers now move in the opposite direction.

- **Integrated loading and unloading.**
  After a portainer has finished unloading a bay it starts loading it. In this case the portainer does not have to move as much, thereby increasing productivity. The flows of the combined import and export containers make the planning in the stack more difficult.

- **Simultaneous loading and unloading.**
  This method concentrates on the utilisation of the empty portainer spreader as it returns to the container vessel during the unloading procedure. By simultaneously unloading and loading a container vessel in this manner, the occupancy rates of the portainer spreader will increase. Accurate terminal planning is an extremely important condition for the correct implementation of this method.

In the simulation model the aim is to work according to the separate loading and unloading strategy combined with the grazing method.

### 3.5 Terminal strategies

The container flow through a terminal is a complex series of interconnecting activities. The output of the whole system depends on the capacity of the weakest link in that chain of activities. Basically a transhipment container terminal consists of three terminal components. The first component is the portainer, which is responsible for the loading and unloading of the vessels. The second component is the horizontal transportation, which takes the containers from the quay to the yard. The horizontal transportation directly influences the unloading and loading of the vessel. The container storage is the third component in the system.

In order to avoid congestion of the horizontal transportation units the production of the stack cranes must have a production comparable to the portainers. As this study involves the design of a new terminal, the container storage system can be favourably placed directly behind the stacks. Figure 3.3 gives a schematic and simplified representation of a container terminal, which is the basis of the design and simulation study.

![Figure 3.3, schematisation of transhipment terminal](image)
As this study is based on the design of a transshipment terminal, containers can only enter or leave the terminal by vessel.

**Berthing strategy**

As stated before the terminal will be visited by two kinds of vessels, a regional feeder vessel and an intercontinental mega vessel. The three available berths have been divided into two feeder berths and one mega vessel berth.

The most efficient configuration of the berths is to reserve the middle berth, Berth 2, for the mega vessels. This way the relative travelling distances for the horizontal transportation units will be minimised.

![Diagram of Berthing Strategy](image)

*Figure 3.4, a three berth terminal*

**Container handling strategies**

An important aspect of terminal design is determining the most efficient way of transporting the containers to and from their right locations in the stacks during the unloading and the loading of the vessel. A terminal with three or more berths can achieve considerable advantages by determining a favourable container handling strategy on the terminal. Especially when the stacks can be placed directly behind the berths as is the case in this study.

Considering the established berthing strategy, two relevant handling strategies will be discussed resulting in the container handling strategy that will be applied in this study. A consequence of the hub transshipment assumption and the berthing strategy is that feeder vessels mooring at Berth 1 or Berth 3 bring in all the regional containers to the terminal. These containers will stay in the stacks until they are eventually loaded onto a mega vessel moored at Berth 2. The opposite also applies. Containers brought in by mega vessels will eventually be loaded onto feeder vessels moored at Berth 1 or Berth 3. This naturally has consequences for the container handling strategies on the terminal.

Basically two different vessel unloading strategies can be applied, each resulting in a typical vessel loading strategy.
Fast unloading /adjacent loading strategy
If the unloaded containers are stacked directly behind the unloading berth, the travelling distances for the horizontal transportation units will be limited. Another advantage is that the encounters between horizontal transportation units operating on other berths and stacks are minimised. However, if for example a feeder vessel moored at Berth 1 unloads all its containers in Yard 1, the loading procedure of these containers onto a mega vessel moored at Berth 2 implies horizontal transportation from Yard 1 to Berth 2. This is called adjacent loading.

![Diagram](image)

*Figure 3.5, the fast unloading strategy/adjacent loading strategy*

Adjacent unloading /fast loading strategy
Obviously the strategies can also be inversed. Applying adjacent unloading on the three berths will position the containers in the favourable stacks and facilitate the loading procedure.

![Diagram](image)

*Figure 3.6, adjacent unloading strategy and fast loading strategy*

Including the shuffle procedure to the container handling cycles.
Another container handling strategy is the fast unloading and the fast loading combination. This can only be achieved if the containers have been repositioned from the yard behind the unloading berth to the yard behind the loading berth during a separate procedure called shuffling.

By including a shuffle procedure to the container handling cycles, the containers can be unloaded and loaded very efficiently. From vessel service times point of view this is the most efficient strategy. But it is only possible if there is time available for shuffling. It also demands extra storage capacity in the stacks.

When the shuffle procedure is included, the containers that have been unloaded from the feeder vessels, moored at the berths in front of Yard 1 and Yard 3, will be shuffled to Yard 2 after the feeder vessel has left, ready to be fast loaded onto a mega vessel. Containers
unloaded from mega vessels moored at Berth 2 into Yard 2 will be shuffled to either Yard 1 or Yard 3 ready to be fast loaded onto the feeder vessels.

Figure 3.7, the fast loading, fast unloading and shuffle strategies
3.6 Arrival cycles of the vessels

Although this study will concentrate on the processes taking place at the terminal a virtual vessel arrival cycle has been assumed in order to make calculations concerning the required stack capacity.

In order to have a consistent simulation analysis the vessel types included in the simulation program have been limited to the two types described above. Of course, in reality a variety of vessels will be able to attend the terminal. The arrival cycles of the two types of vessels included in the simulation are partially determined by the demands dealt with above and the following boundary conditions.

- The occupancy rate of the terminal and the berths may not exceed 60 %, and should preferably be around 50 % in order to allow variations in arrival times without creating large congestion.
- The time needed to unload and load the vessels may not exceed 24 hours. This is including time needed to moor the vessel.
- The TEU-factor is estimated to be 1.5. This implies that the amount of 20’ containers that is handled on the terminal is the same as the handled amount of 40’ containers.
- There are separate berths for feeder vessels and for intercontinental vessels. In this study this has been assumed to be one berth for the mega vessels and two berths for the feeder vessels.

The pre-simulation calculation of the arrival cycles of the vessels can be seen in Appendix 3. The main results are given below.

<table>
<thead>
<tr>
<th>Vessel type</th>
<th>Vessel capacity</th>
<th>Parcel size</th>
<th>Loading &amp; unloading</th>
<th>Arrival cycles</th>
<th>Hours available per vessel</th>
<th>Portainers per vessel</th>
<th>Moves per portainer per hour</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mega vessels</td>
<td>8000 TEU</td>
<td>4725 TEU</td>
<td>9450 TEU</td>
<td>2 calls in 3 days</td>
<td>17.5</td>
<td>6</td>
<td>60</td>
</tr>
<tr>
<td>Feeder vessels</td>
<td>1600 TEU</td>
<td>1600 TEU</td>
<td>3200 TEU</td>
<td>2 calls in 1 day</td>
<td>8.75</td>
<td>4</td>
<td>60</td>
</tr>
</tbody>
</table>

*Table 3.1, Results of cycle times calculations*

The average required capacity of the portainers working on feeder and mega vessels is assumed 60 moves per hour as will be justified in the next chapter. The result of this calculation is that the berth occupancy rate of the mega vessel berth, Berth 2, is around the 54 %, while the occupancy rate of the feeder vessel berths, Berth 1 & 3 is around 45 %.

As follows from this vessel arrival scheme, the annual container throughput of the terminal will be around 4.6 million TEU. Per berth the annual throughput will amount to over 1.5 million TEU. If this amount is to be approached during simulation, the horizontal transportation and the container storage system must be able to process the production of the portainers. This configuration of ship arrival cycles can be seen as a benchmark for the terminal and will be checked at the end of this report.
3.7 Required container storage capacity

The method used to calculate the storage capacity needed in the stack takes the arriving cycles of the container vessels into account. Calling intervals of vessels and quantity of containers handled per vessel call are now introduced in the calculation. To calculate the stack capacity for transhipment terminal in this manner Watanabe [L.12] introduced the following formula:

\[
C_0 = q_u + p_u(D_u - p)/D_u + q_u(D_u - 2p)/D_u + \ldots + q_u(D_u - np)/D_u \\
+ q_f + q_f(D_f - (p-1))/D_f + q_f(D_f - (2p-1))/D_f + \ldots + q_f(D_f - (np-1))/D_f
\]

With:
- \( q_u \) = number of TEU to be unloaded from a mega vessel
- \( q_f \) = number of TEU loaded onto a mega vessel
- \( D_u \) = number of days that the feeder vessels are allowed to deliver containers in advance for the mega vessel
- \( D_f \) = number of days that the feeder vessels are allowed to retrieve containers after the arrival of the mega vessel
- \( p \) = periodic interval of mega vessels in days
- \( n = 1, 2, 3, \ldots \)

Assumptions that have been made

- By simplifying the model to two kinds of ships, the 8,000 TEU mega container vessel and the feeder vessels, an initial storage calculation can be made.
- Assuming that the berth capacity of the terminal should be maximised within the boundaries of a 50% berth occupancy rate, the arrival scheme of the 8,000 TEU vessels is once every 1.5 days. Feeder vessels arrive twice a day.
- The feeders are allowed to deliver containers starting from 5 days in advance.
- The share of the terminal on the mega vessels is assumed to be 60%, resulting in 4,800 TEU to be unloaded of and the same number to be loaded again.
- Feeder vessels provide these containers resulting in 1,600 TEU to be unloaded and the same amount to be re-loaded per call.

Calculations

When the feeders arrive once every 1.5 days, and the deliveries for a mega container vessel start 5 days in advance the required stack capacity can be calculated.

Using the Watanabe formula the required capacity is: 24,000 TEU

This calculation has been carried out in Appendix 4.
4 Choice of the terminal components and their capacities

The traditional terminal activities can be split up into three main components. The first component is the portainer that is responsible for unloading and loading the containers from and onto the vessels. The second component, the horizontal transportation, receives containers from the portainers during the unloading of a vessel and provides the portainers with containers during the loading of a vessel. Containers are stored in stacks. This can be seen as the third terminal component.

In the following chapters each of these terminal components will be discussed. The portainers will be analysed by introducing the portainer handling capacities. In this study the containers will not be conventionally stacked upon each other but will be stored in racks. Following the discussion, the most promising warehousing concept will be explained and improved for the specifications of this terminal. For the horizontal transportation component two alternatives will be compared using the two most important criteria in this study: land usage and production capacity.

4.1 Portainers

Along with the development of mega container vessels, crane designers have started to develop super productive portainers. The growth of production capacity of the conventional single hoist portainers by increasing trolley speed and acceleration resulting in a heavier framework seems to have reached its limits.

In the last couple of years new concepts have been introduced that should be able to increase crane productivity. Portainer productivity however must be seen as an integral part of the terminal system. After all, super productive portainers require super productive terminals.

Some of the concepts are:

*Dual hoist system, fixed platform [L.5] & [L.9].*

Splitting the trolley cycle into two processes and introducing a buffer on the fixed platform can significantly increase portainer productivity.

![Diagram of Dual hoist system with a fixed platform]

*Figure 4.1, Dual hoist system with a fixed platform*
Dual hoist system, elevating platform [L.8].
Lowering and elevating the platform of the portainer will decrease the travelling distance of the spreader thereby increasing portainer production. An independent simulation study described in [L.8] shows an average container production of 54 container moves per hour.

Figure 4.2, dual hoist system with elevating platform

Double trolley system [L.6].
Handling a container vessel with two trolleys operating independent of each other promises considerable handling productions. Average productions varying from 55 – 82.5 container moves per hour according to rough calculations.

Figure 4.3, double trolley system

Twin-lift equipment / spreader [L.7].
By introducing twin-lift equipment the amount of TEU per move will increase from an average of 1.5 TEU/move to 2 TEU/move.
Elevating girder cranes [L.8].
By allowing the girder of a portainer to elevate and to lower along with the height of the stacked containers on a vessel the hoisting height is decreased, improving handling production.
[L.4] shows an average production varying from 55 container moves per hour to 66 container moves per hour.

*Figure 4.4, elevating girder crane*

Ship trolley/shuttle/shore trolley system (Paceco²).
Paceco [L.9] claims to have the most productive concept with a maximum average production of up to 90 moves per hour on a hatchless Panamax ship. An independent study by [L.4] claims an average production of 67 moves per hour.

*Figure 4.5, Paceco supertainer*

The supertainer consists of three independent operating container carriage devices. During unloading the seaside trolley retrieves a container from a bay in the vessel and delivers it to the shuttle. The seaside trolley only makes a vertical movement while the shuttles carry out the horizontal movements. The landside trolley is responsible for the vertical movement necessary to deposit the container onto the horizontal transportation units.

The development of portainers in the last couple of years has shown a considerable increase in production. In this study no particular concept has been chosen, only an assumption on portainer productivity has been made. How the portainer productivity is modelled in the computer simulation will be explained in the next chapter.

² Paceco cooperation, the inventor of the Supertainer™
Determination of the portainer cycle times

In this study portainer cycle times from FAMAS [L.12] have been used. They are derived from a logistic analysis using a hatchcover vessel and a portainer called "jumbo container crane". The course of the portainer cycle times in time is given in Figure 4.6.

![Portainer cycle times when unloading and loading a Feeder](image)

These are the portainer cycle times when 4 portainers are unloading and subsequently loading a feeder vessel. Moves 1 to 1266 represent the unloading of the feeder vessel implying 1266 containers will have to be unloaded. Operating with 4 portainers means each portainer will unload approximately 315 containers.

The unloading starts with the container located on the position closest to the landside of the vessel. The time needed to retrieve this container is relatively short, as can be seen in Figure 4.6. After the first container has been unloaded, the portainer works horizontally towards the seaside of the vessel, unloading the top layer of containers. After the first layer has been unloaded the portainer takes away the second layer. This unloading process of one layer creates a tooth like shape in the portainer cycle times. In total there are 9 layers in this feeder vessel, which amounts to 9 teeth in the function. The cycle times of the successive layers will become higher as the portainer reaches deeper inside the vessel.

After the first hold has completely been unloaded the portainer will transfer to the next hold and start all over again. Each portainer has 3 holds to handle as can be seen in Figure 4.6.

Moves 1267 to the end represent the loading of the vessel. During the loading of the feeder vessel the portainers will first deposit a container on the bottom of a hold at the seaside of the feeder vessel and work their way back to the landside of the vessel.

The average cycle time for the unloading and loading is 58 seconds, which could lead to a handling production of 62 moves per portainer per hour.
4.2 Container storage system

As land usage needs to be minimised, stacking height is an important design factor. Normally containers are stacked on top of each other. This conventional stacking method is restricted to nine containers in height by ISO standards. Apart from the stacking height restrictions, there is another disadvantage to conventional stacking. When containers are stacked on top of each other shuffling and reshuffling of containers is a time consuming process, and takes more time with increasing stacking height.

Placing containers in "racks" or "warehousing" containers is a new approach in container storage. The big advantages are that stacking heights are not restricted to ISO limits, and even more important it reduces the time consuming shuffling. This random container access greatly decreases the retrieval and depositing time of a container in a stack. This subsequently permits quicker loading and unloading of container vessels. Another advantage is the reduction of labour cost associated with the handling of containers in the stack, as the system is completely automated. A disadvantage could be the high investment cost to be made for the expensive construction and foundation.

Momentarily several variants of container warehousing are being researched. For this study the most relevant and credible one [L.14] will be described in the following chapter. Using this container-warehousing variant as a basis, some alternations have been introduced in order to improve the production capacity.
4.2.1 Description of the chosen warehousing variant

The container warehouse as described in [L.14] consists of the following elements:
- overhead portal cranes
- storage racks
- stacking cranes called "elevating transfer vehicles" in [L.14]

Below, these different components of the warehouse will be discussed individually and in relation with each other. The bold numbers between the brackets used in the text refer to numbers used in the different figures.

![Diagram of warehouse](image)

*Figure 4.7, cross-section of a warehouse*

*Unloading/loading area*

The overhead portal cranes [1] are responsible for the loading and unloading of the horizontal transportation units [2] as shown in Figure 4.7. The overhead portal crane consists of a rail-guided carriage supporting the hoist frame that holds a spreader. The spreader has twist locks, which engage the four fittings located on the top of the container. The portal crane is movable between positions above the portal buffer area [3] and the loading/unloading zone [4] of the horizontal transportation vehicle [2].

As can be seen in Figure 4.8 these portal cranes [1] are distributed alongside of the warehouses. The horizontal transportation units [2] can also be distinguished. Container length [6] has a great influence on the warehouses and on the elevating transfer vehicle [5].

![Diagram of warehouse](image)

*Figure 4.8, top view of a warehouse*
Storage rack
The storage rack as shown in Figure 4.9 is the framework that holds the container [7]. This rack consists of a system of vertically and horizontally orientated metal beams [8] forming several container storage bays. Each bay is provided with supporting brackets [9] for supporting the container at its four lower corners. The bays have a depth sufficiently large to store two containers. This could make occasional shuffling necessary.

Figure 4.9, the storage rack of a warehouse
Elevating transfer vehicles
To get a container into a stacking bay, elevating transfer vehicles are used [5] as shown in Figure 4.9. These vehicles take care of the vertical transportation and, when needed, the horizontal transportation in the warehouse. An elevating transfer vehicle comprises of a stiff framework attached to a rail mounted support vehicle. In the framework a cradle [10] can be lifted and lowered as desired. The cradle contains a shuttle [11] that can be moved laterally in and out of the cradle to extend into a bay in the storage rack or overhead portal crane area.

For example: as the elevating transfer vehicle arrives at the storage rack, the shuttle leaves the cradle and enters a bay containing a container [7]. Having located itself under the container, the shuttle lifts the container from the four lower corners supporting devices [12]. The shuttle returns to the cradle of the elevating transfer vehicle [10], which will move itself to the desired new container location where the process will take place in reverse order.

Figure 4.10, front view of the lifting process of a shuttle

In Figure 4.10 the lifting process of a shuttle is shown. The shuttle [11] releases the container [7] from its four supporting devices [12] by lifting it with a lifting device [13].

In Figure 4.11 the cradle [10] of the elevating transfer vehicle is shown. The container [7] is supported by the shuttle of the elevating transfer vehicle [11], which in turn is supported by the underside of the cradle [6].

Figure 4.11, the cradle of the elevating transfer vehicle
4.2.2 Adaptations to the chosen warehousing concept

In the previous chapter the chosen container-warehousing concept has been described. The choice was based on the random container access and the absence of the container stacking height limits posed by the ISO standards. However, there are some apparent shortcomings. These shortcomings will be described and addressed in this chapter.

- Stack production
  As can be seen in the top view in Figure 4.8 the number of portal cranes and unloading/loading platforms [1] are limited. As a result, bottlenecks might appear during the handling process.

- Separation of functions on the ground level
  There are four functions present on ground level as can be seen in Figure 4.7. The first function is horizontal transportation consisting of horizontal transportation lanes [4] and horizontal transportation units [2]. The second function is formed by the unloading/loading areas where the portal cranes [1] operate. The third function is the area reserved as the operating lanes for the elevating transfer vehicles. This arrangement will create problems at possible crossing points between horizontal transportation units and elevating transfer vehicles. The fourth and last function on the ground level is container storage. As the necessity for container storage on the ground floor is limited, this area might have a better application. Container storage bays might just as well start at the second floor.

- Land usage
  As land usage is to be minimised in this study, some additional adaptations have to be made. In the container storage area on the container terminal, the object is to design container storage bays without unused area on the higher floors. Currently, the area above the horizontal transportation lanes [4] is left unused as can be seen in Figure 4.7.

The adaptations applied to the container warehousing system necessary to solve the problems posed above are discussed in the following.
Unloading/loading area
The changes in the warehousing concept are most apparent for the unloading/loading area as can be seen in Figure 4.12. By moving the container storage bays to the second floor of the warehouse a great deal of space can be gained on the ground floor. This enables the placement of two extra horizontal transportation lanes [14] to be placed thereby considerably increasing the stack production.

The most striking difference however, is the absence of the portal cranes. The portal cranes have been replaced by the so-called service points [15]. A service point consists of one container location located upon an elevated area with the same height as a horizontal transportation unit [2]. In order to make the lateral horizontal movement during the unloading or loading of a container upon a horizontal transportation unit, each service point is provided with a shuttle [11] comparable to the shuttle of the elevating transfer vehicle.

![Figure 4.12, a cross section of two warehouses](image)

In Figure 4.13 the top view of the adapted warehouse can be seen. The two extra horizontal transportation lanes [14] can be distinguished as well as the locations of the service points [15]. The service points are all coloured grey and are distributed all along the horizontal transportation lanes. This will secure a great increase of container handling production.

![Figure 4.13, top view of the warehouse with adaptations](image)

The loading procedure of a container located on the shuttle in the cradle of the elevating transfer vehicle onto a service point is as follows: when the elevating transfer vehicle arrives next to the service point, the shuttle in the cradle containing the container transfers lateral onto the service point. The cradle shuttle lowers the container onto the service point shuttle and returns back to the cradle allowing the elevating transfer vehicle to proceed with its next task. As a horizontal transportation unit arrives at the service point requesting a container, the service point shuttle lifts the container from the supporting devices and transfers lateral upon the horizontal transportation unit where it lowers the container onto supporting devices. The shuttle then transfers back onto the service point and waits for the next task.
The great advantage of this system is that a horizontal transportation unit can be serviced while an elevating transfer vehicle is still busy on an other job. This increases the flexibility of the system.

_Elevating transfer vehicle_

In Figure 4.12, two possible types of elevating transfer vehicles are shown. The rail mounted crane and the hanging crane. The ground floor has been kept free in order to allow the crane to cross any horizontal transportation lanes, which increases the flexibility of the concept. Of the two described crane possibilities the rail-mounted variant seems be in favour. A hanging crane could also be a possibility, but it will be much harder to make the construction stiff and rigid. This is necessary to avoid time loss due to sway problems.
4.3 Horizontal transportation

There are numerous horizontal transportation concepts. The choice of a horizontal transportation concept largely depends on the type of terminal that is to be designed. A terminal with relative low container throughput and a high availability of terminal area will have an other optimal horizontal transportation concept than a terminal with relative higher container throughput and a lesser availability of terminal area. The choice also depends on the suitability to work together with the two other terminal components portainer and container storage.

The terminal in this study will be designed as an automated high production terminal while minimising the terminal area. The horizontal transportation concept will have to comply with these specifications resulting in the following demands. The horizontal transportation units have to preferably:

- be completely automated
- be able to process large amounts of containers
- occupy a minimum of land area

Taking the automation demand into account the options are limited to automated guided vehicles. The most credible ones are:

- Rubber tired automated guided vehicles (agv’s)
- Rail guided automated guided vehicles (rgv’s)

In the next sections the both horizontal transportation concepts will be illustrated and a choice between them will be made.
4.3.1 Rubber tired automated guided vehicles (agv’s)

Agv’s have already been fully operational for over a longer period of time on the ECT-terminal in the port of Rotterdam. Due to the operational experience, the agv has gained in reliability. Within the FAMAS-study, research has also been carried out on improvements of the ECT-agv’s [L.11], resulting in the FAMAS agv.

Driving force
Each agv has a hydrostatic diesel engine that is responsible for the propulsion. The agv performance is the same for normal and reverse driving. The disadvantages of this kind of propulsion are the refuelling and maintenance stops. Refuelling stops are scheduled once every 80 hours during normal operation [L.12].

Navigation
The navigational system at the ECT terminal consists of a line grid of inductive wires orientated parallel and perpendicular to the quay wall, called the Free Ranging on Grid (FROG) system [L.10]. Sensors detect possible deviations in the path of the agv. The deviations are then registered and compensated by the agv. In this manner a positioning accuracy of ± 3 cm can be achieved.

Operational properties
In table 4.1 the properties of the ECT-agv and of the FAMAS-agv are given [L.12]

<table>
<thead>
<tr>
<th>Operational properties</th>
<th>ECT-agv</th>
<th>FAMAS-agv</th>
</tr>
</thead>
<tbody>
<tr>
<td>Speed loaded (m/s)</td>
<td>5.5</td>
<td>7.0</td>
</tr>
<tr>
<td>Speed unloaded (m/s)</td>
<td>4.0</td>
<td>7.0</td>
</tr>
<tr>
<td>Acceleration loaded (m/s²)</td>
<td>&gt;0.3</td>
<td>0.5</td>
</tr>
<tr>
<td>Acceleration unloaded (m/s²)</td>
<td>&gt;1.0</td>
<td>&gt;1.0</td>
</tr>
<tr>
<td>Speed in turns (m/s)</td>
<td>3.0</td>
<td>&gt;3.0</td>
</tr>
</tbody>
</table>

Table 4.1, operational properties of a rgv

As land usage is an important design parameter in this study, the required turning radius of an agv must be minimised. A small turning radius can only be established with low agv speed in order to achieve the maximum wheel turning. Wheel turning speed of the inside wheel of an ECT-agv is 15° per second. With the agv speed decreasing to 3 m/s the turning radius is 18 meters. The time needed to complete a 90 degree turn with this configuration is 9.5 seconds.

Figure 4.14, agv turning radius
4.3.2 Rail guided vehicles (rgv's)

Another system for horizontal transportation is rail guided agv's or rgv's. Preussag Noell is developing this system in Germany and in the People's Republic of China [L.13]. Based on technology and experience acquired during the exploitation of the transrapid magnet trains, the rgv concept has been specially adapted for high performance terminals. The layout of the rails consists of runways orientated parallel and perpendicular to the quay suitable for longitudinal and lateral displacements of rgv's. The wheel sets of the rgv can rotate trough 90° angles as can be seen in Figure 4.15.

![Figure 4.15, rgv taking a turn](image)

The runway consists of ordinary UIC 60 rails mounted on steel twin sleepers. The rails must be accurately laid to guarantee a gap between magnets and the stator of 7 ± 3 mm. The guide system consists of three modules: a motion for transversal transition, a module for longitudinal transition and the crossings where the two directions meet. To make it possible to turn all four wheel sets of the rgv's, four circular steel discs with transverse guides have been fitted at the crossing points. A hydraulic power unit turns the discs individually but simultaneously.

*Driving force*

The rgv's are driven by means of contact free linear synchronous motors, which are distributed over the layout as required for a suitable driving force. They act on the permanent magnets located on the underside of the rgv. It is possible to set a variable speed by means of a mobile electromagnetic field, generated by a frequency converter.

*Navigation*

A contact free actual detecting system is integrated into the runway, responding to the individual magnets located on the rgv. This allows to determine the position of the rgv and to supply the input values required to ensure that the linear drivers are supplied with power and switched over in the correct order. The claimed position accuracy is ±3 mm.
Operational properties
The operational properties of the rgv during tests on the demonstration plants of Preussag Noell have been taken from [L..13] and are shown below.

<table>
<thead>
<tr>
<th>Operational properties</th>
<th>rgv</th>
</tr>
</thead>
<tbody>
<tr>
<td>Speed loaded (m/s)</td>
<td>3</td>
</tr>
<tr>
<td>Speed unloaded (m/s)</td>
<td>3</td>
</tr>
<tr>
<td>Acceleration loaded (m/s(^2))</td>
<td>1</td>
</tr>
<tr>
<td>(Assumed property)</td>
<td></td>
</tr>
<tr>
<td>Acceleration unloaded (m/s(^2))</td>
<td>1</td>
</tr>
<tr>
<td>(Assumed property)</td>
<td></td>
</tr>
<tr>
<td>90°-wheel rotation (s)</td>
<td>4</td>
</tr>
<tr>
<td>(Assumed time)</td>
<td></td>
</tr>
</tbody>
</table>

The rgv has a length of 15 meters and is 2.8 meters wide.

4.3.3 Criteria and choice

The choice which horizontal transportation concept to use for the design of the terminal is based on a number of criteria. These criteria will be discussed in this section.

Operational properties
Operational properties like speed and acceleration will not be the most important criteria for the choice of the horizontal transportation system, as the properties are more or less comparable for both horizontal transportation units. Especially because the terminal will be designed as a compact terminal driving distances will be limited.

The time needed to complete a 90° turn is comparable for both alternatives. A rgv travelling at a speed of 3 m/s consumes 10 seconds for deceleration (3 s), wheel turning (4 s) and acceleration (3 s) while the agv consumes 9.5 seconds for completing the turn.

Land usage
The land usage of the horizontal transportation vehicles is an important planning factor. During the taking of the turns the land usage of the rgv's is more efficient than that of the agv's. This results in a better use of the area on the terminal. The land usage of the horizontal transportation units is the key choice factor in this study as land usage an important objective.

The preferred horizontal transportation unit type for the design of the terminal is the rgv. The most important criterion is the usage of terminal area.
5 The design of the terminal Layout

In this chapter the layout of the container terminal will be established. The layout consists of
rgv lanes, portainers and stacks. The realisation of the layout will be reproduced step by step
thereby giving an impression of all the design considerations that have been made throughout
the designing process.

First, the capacity and the positioning of the container warehouses will be dealt with, followed
by the positioning of the two types of rgv lanes in the layout. Some preliminary layout
calculations have been made in order to get a first impression on the capacities of the
terminal.

5.1 Positioning of the container warehouses

As the terminal will not be implemented in a specific topographical environment, the
placement of the container warehouses will be assumed not to be affected by geological
restrictions. The warehouses will therefore be placed directly behind the quay containing the
portainers.

As stated in Chapter 3, the quay of the terminal will contain three longitudinal orientated
adjoining berths. By placing the warehouses directly behind the portainers, the orientation of
the warehouses will be determined by the properties of the rgy's. The warehouses have been
designed in such a way that rgy's can only enter in a straight on fashion. The consequence
for the warehouses is that they are to be orientated parallel to the quay as can be seen in
Figure 5.1.

The next step in the design process is the determination of the required storage capacity per
yard behind each berth, for the pure transhipment terminal. The capacity calculation has been
carried out by assuming arrival cycles of feeders and mega vessels.
Based on the arrival cycles of the vessels and the amount of containers unloaded and loaded
per call, the needed total storage capacity of the three yards has been calculated in Chapter
3.

The required container storage capacity is 24,000 TEU. The storage capacity will be spread
out evenly over the terminal. Berth 1 and Berth 3 both have a length of 250 meters while
Berth 2 has a length of 370 meters. Proportionally Berth 1 and Berth 3 must each be able to
host an amount of 6,860 TEU, while Berth 2 should be able to host an amount of 10,290 TEU.
Based on Appendix 5 the number of warehouse rows has been calculated to amount to 5, and will be placed over the length of the terminal. By choosing 5 warehouse rows, the stacking height will amount to 12 stories of warehouse racks. Adding the ground level means that the warehouse will be 13 stories in height. It is assumed that it is possible to construct a feasible container warehouse with this height. However, it is recommended to conduct a study on this subject.

The length of a container has great influence on the dimensions of the storage bays and the warehouse crane. For this reason 20’ container stacks have to be separated from the 40’ container stacks. Having assumed a TEU factor of 1.5 container storage locations must be consistent with this relation. This means that the number of 20’ container bays and 40’ container bays should be the same. As one 20’ container warehouse contains twice as much storage bays as a 40’ container warehouse, the ratio 20’ warehouses − 40’ warehouses should be 1:2. Having implemented 5 container warehouses, the best approach to the 1:2 ratio is by reserving 2 warehouses for 20’ containers and 3 warehouses for 40’ containers. Warehouse rows 1 and 2 will therefore be 20’ container warehouses, while warehouse rows 3 to 5 will be designed as 40’ container warehouses.

5.2 Positioning and capacity of the rgv lanes

Having established the properties of the yard in the previous section the layout will now be expanded with the rgv lanes. Besides the warehouse lanes inside the warehouses, yard-lanes and inter-berth lanes can be distinguished. Yard-lanes are orientated perpendicular to the quay and guide the rgv’s coming from outside the yard into the yard. Inter-berth lanes are orientated parallel to the quay and maintain the connection between the berths. Placing these lanes on the terminal will form a grid of lanes. In the following chapters the required capacity of both lane types will be calculated followed by the placement of the lanes.

5.2.1 Inter-berth lanes

In order to calculate the required number of inter-berth lanes the capacity of one single lane must be known first. In Appendix 6 the capacity of an inter-yard-lane has been determined to be about 4 rgv’s per minute or one every 14.3 seconds. The normative situation to be used for calculating the needed lane capacity will be adjacent loading on all three berths. This means that two feeder vessels moored at Berth 1 and Berth 3 and a mega vessel moored at Berth 2 are being loaded as explained in Chapter 3.5. Figure 5.2 shows the schematised situation.
Figure 5.2, schematisation of inter-berth rgv flow when three vessels are being adjacent loaded.

When a mega vessel is adjacent loaded, portainer production is in average one per minute. As the containers required for the loading on Berth 2 are assumed to be equally distributed over Yard 1 and Yard 3, 3 containers per minute should be provided from these two adjacent berths to the portainers located on Berth 2. The same amount of empty rgv's will leave the portainers to collect a next container. This means that 3 rgv's will pass both borders of Yard 2 in two directions every minute.

The rgv's working for the portainers located on Berth 1 and Berth 3, will all collect their containers from Yard 2 during adjacent loading. Four rgv's a minute will pass the borders between Yard 1 and Yard 2 in both directions. The same amount will pass the borders between Yard 2 and Yard 3 every minute. Adding up the inter-yard container flows shows a needed rgv capacity of 7 containers per minute in both directions and on both boundaries. This results in 14 rgv boundary crossings per minute.

The capacity of an inter-yard-lane is 4 rgv's per minute as is illustrated in Appendix 6. With 4 inter-berth lanes the inter-berth transportation capacity is about 16 rgv boundary crossings a minute. This seems to be enough, but during the simulation phase the rgv flows will probably become more irregular. Applying more inter-berth lanes will increase the flexibility of the inter-berth transportation. This is the reason that the simulation study will be carried out with 8 inter-berth lanes as can be seen in Figure 5.3. Later on a simulation run with 4 inter-berth lanes will be carried out to check this assumption.

Figure 5.3, Layout of Berth 2 including inter-berth lanes
Arrows represent the lane directions. The portainer service lanes have been placed behind the portainers. For now it is assumed that two will suffice. The computer simulation will give verification of this assumption.

5.2.2 Yard-lanes

The rgv lanes that make the warehouses accessible are called the yard-lanes. These lanes connect the warehouse lanes to the inter-berth lanes and the portainer service lanes. The yard-lanes will be placed perpendicular to the quay in the layout. The number and distribution of the yard-lanes depend on the required lane capacity. The lane capacity of the yard-lane is calculated in Appendix 6 and about 4 rgv’s per minute.

Having established the yard-lane capacity makes it possible to place the yard-lanes in the layout. The average portainer production is again assumed to be 1 container per minute. This implies that during the unloading/fast-loading of a vessel, four portainers can be connected to one yard-lane. However, if during the unloading/fast-loading of a vessel another vessel is being adjacent loaded, the needed capacity of the yard-lane will increase dramatically. For this reason the yard-lanes have been designed to be supplied by two portainers. Whether or not the yard-lane has enough capacity to be handled by two portainers, has been checked in Appendix 7 by performing a performance calculation for the entire terminal.

During the unloading/fast-loading of a vessel, the rgv’s cover a loop between the portainers and the stacks located in the yard. There are two distinctive variations for the placement of the yard-lanes. These two variations will be discussed now.

Opposite rotation rgv loops
In this case two adjacent portainers have opposite orientated quay-stack loops, as can be seen in Figure 5.4. As one yard-lane can handle a capacity of at most two portainers every portainer is enclosed by a rgv loop. This way a tight grid can be formed. Each warehouse lane now holds three 40’ container or six 20’ container service-points. The warehouse lanes are capable of hosting a maximum of 3 rgv’s.

In order to make it possible to shuffle containers more efficiently between berths, half of the warehouse lanes located behind a portainer have been given an opposite direction to the described unloading loop. Rgv’s carrying out shuffling orders can also enter the stack from the above-situated inter-berth lanes.

Figure 5.4, Layout version 1 including warehouse lanes
Equal rotation rgv loops

As is the case in the previous layout, one warehouse lane connects two adjacent portainers to the stacks located behind them. The circular motion of the rgv's is now equally orientated instead of opposite resulting in the layout designed in Figure 5.5.

![Image of Quay stack loops with equal orientation](image)

*Figure 5.5, layout version 2 including warehouse lanes*

The number of 40' container service points per warehouse has increased to 6, while the number of 20' container service points has increased to twelve per warehouse. The grid of the version 2 layout is less tight than the version 1 layout, resulting in a higher chance of congestion. Another problem might appear due to the fact that the yard lanes are placed right next to each other. Portainers located above the two yard-lanes might have difficulties in reaching the rgv's waiting in their queue while handling a vessel.

The tighter grid of version 1 combined with the possible handling problems of the portainers in equal rotation are the reasons the computer simulation will be carried out with the opposite loops layout.

In Appendix 7 the overall needed capacity of the established layout has been calculated for the most extreme situations. The capacity seems to be sufficient on all the lanes. The computer simulation will give a more definitive answer.
5.3 Additional terminal design aspects

Some terminal aspects have not yet been included in the terminal layout and will be briefly clarified in this section.

Rgv-parking
Not all available rgv's will be active constantly. For this reason a rgv-parking has been added to the layout. The positioning of the parking has not been extensively examined, but can be positioned behind the stacks past the inter-berth lanes, as can be seen on Figure 5.7.

Hatches of the container vessels
A number of container vessels are equipped with hatches that separate the containers located in the hull of the vessel and containers located on top of the vessel. These hatches need to be removed and placed on the quay during the handling of the vessel. A space has been reserved behind the portainers.

Elevated rear rails of the portainers
In this study only the processes on the terminal will be simulated during the handling of container vessels. A result of not including vessel arrival schemes makes the determination of the ideal number of portainers impossible. For this reason the layout drawings of the terminal always include the maximum number of portainers. In a later phase however an optimum number of portainers should certainly be determined. This probably means that the number of portainers will be less than the maximum number. This implies that portainers have to be able to transfer from one location to another. Yard lanes will have to be crossed during this process. As the rail mounted hind legs of the portainers may not cross the rgv yard lanes the hind legs of the portainer will be elevated to the first floor allowing the rgv's to pass underneath.

![Diagram of container warehouse and portainer](image)

*Figure 5.6, side view of the portainer*
Service lanes and off-standard lanes
Service lanes will be placed in front of the portainer, while the off-standard lanes are placed behind the front legs of the portainers.

In the previous sections of this chapter the evolving of the layout of the terminal has been described one step at a time. Integrating the chosen framework of rgv lanes, the container storage and the portainers on their portainer locations, will give a complete overview of the terminal. In Figure 5.7 all the items on Berth 1 have been named.

![Diagram](image)

Figure 5.7, Berth 1

The complete terminal layouts have been included in the appendices. The first layout as show in Appendix 1 contains the dimensions, while the second layout, shown in Appendix 2, shows the numbers given to all the nodes. The numbers of the nodes will be used to simulate the terminal.
5.4 Handling strategies applied to the terminal Layout

The handling strategies as described in Chapter 3 will now be applied to the chosen layout. The chosen handling strategies for the three berths consist of unloading, fast loading, adjacent loading and shuffling. In this order the strategies will be dealt with in this chapter.

5.4.1 The unloading of a container vessel

When a container vessel is being unloaded the containers will always be delivered to one of the stacks located in the stack-clusters behind the unloading portainer. This unloading strategy is the same for feeder vessels moored at Berth 1 or Berth 3 and for the mega vessels moored at Berth 2. The rgv's delivering the containers from their portainer to a stack will always drive in the same circular motion. The direction of this circular motion depends on the location of the unloading portainer. A consequence of the fixed direction of the circular motion is that the unloading rgv's can only enter 5 of the 10 warehouse lanes of a stack-cluster, as these lanes are one directional, as can be seen in Figure 5.8.

![Figure 5.8, unloading strategy of a container vessel](image)

The unloading rgv's will not enter an inter-berth lane during the unloading process. A consequence of this strategy is that the yard-lanes will be more crowded close to the portainer, as all of the unloading rgv's will have to enter the portainer queues. Closer to the far end of the yard, opposite of the portainer, the yard-lanes will be less crowded as rgv's have already entered warehouse lanes located closer to the portainers. Rgv's arriving from Berth 2 might make use of this.
5.4.2 The adjacent loading of a mega vessel

As opposed to the unloading strategy, which is the same for all berths, the adjacent loading strategy differs per berth. First the adjacent loading strategy of the mega vessels will be dealt with. The mega vessel being handled is always moored at Berth 2. After having been unloaded the mega vessel can be loaded in three different manners. The mega vessel can be adjacent loaded, which means that the to be loaded containers have to be retrieved from the yards in which they have been unloaded into by the feeder vessel berths. A second possibility is that the to be loaded containers have already been shuffled to Yard 2 from the feeder vessel yards, and can be loaded from Yard 2. This is called fast loading. The third option is a combination of the two loading strategies. As the fast loading of a container vessel follows exactly the same pattern as during unloading the fast loading strategy will not be further explained.

Because the feeder vessels provide the terminal with containers for the mega vessels, the adjacent loading strategy of Berth 2 consists of retrieving the to be loaded containers from Yard 1 & 3 and delivering them to the portainers located on Berth 2. The rgv’s will take the inter-berth lanes during this strategy. In order to spread out the rgv flow, it has been split in a rgv flow that make use of the top inter-berth lanes and into a rgv flow that makes use of the bottom inter-berth lanes. In the simulation program the even numbered rgv’s take the bottom inter-berth lanes, while the odd numbered rgv’s take the top inter-berth lanes. Two example routes have been drawn in Figure 5.9, one route for an even numbered rgv and one route for an odd numbered rgv.

![Diagram of adjacent loading from Yard 1 to Berth 2](image)

*Figure 5.9, adjacent loading from Yard 1 to Berth 2*

During the adjacent loading of Berth 2 the to be loaded containers will, of course, also be loaded from Yard 3. As can be seen in the above figure, the rgv’s working for the portainers located on Berth 2 will take the outer “ring” of inter-berth lanes. The inner “ring” is reserved for rgv’s from the feeder berths 1 & 3. This can be seen in the next part of the chapter.
5.4.3 The adjacent loading of a feeder vessel

As is the case with a mega vessel the situation can occur that the terminal has not had enough time to shuffle the containers to the Yards 1 & 3. In that case the containers have to be adjacent loaded instead of being fast loaded. During the adjacent loading of a feeder vessel, the rgv flows will also be split up. An example of two rgv routes for the adjacent loading of a feeder vessel moored at Berth 1 can be seen in Figure 5.10.

![Figure 5.10, adjacent loading of a feeder vessel from Yard 2](image)

During the adjacent loading of the feeder vessel, the rgv’s will take the first “ring” in order to get to Yard 2.

5.4.4 The shuffle process on the terminal

*Shuffling from Yard 2 to Yard 1 & Yard 3*

When a mega vessel has unloaded its containers in Yard 2 the shuffling process on the terminal will begin when the mega vessel has been loaded and has just left the terminal. The rgv’s will not be sent back to the parking but will remain working as shuffle rgv’s. As the portainers of the berth will be inactive during the shuffling process the rgv’s will be subscribed to the stack-cluster located behind the portainers. Each time a container has been shuffled to Yard 1 or 3 the rgv will return to this stack-cluster in Yard 2 in order to retrieve the next shuffle container.

As is the case with adjacent loading for Berth 2, the rgv routes will be split up in two. The top routes will be reserved for the odd numbered rgv’s, while the bottom routes will be reserved for the even numbered rgv’s. The shuffling rgv’s will also take the second inter-berth ring, as is the case during adjacent loading for Berth 2.

A typical shuffle procedure is as follows. A rgv has just delivered one of the last containers for during the loading of a mega vessel. The rgv will be given the assignment to start shuffling. The rgv will drive to a stack behind the portainer with a yard lane orientated in such a way that the rgv can enter the correct inter-berth lane, as it is dependent on its number being odd or
even. After the warehouse crane has loaded the rgv, it will shuffle the container to Yard 1 or Yard 2 where it will enter the right stack. The container will be unloaded from the rgv by the warehouse crane and depending if a container has to be shuffled back to Yard 2 it waits to be loaded again. The rgv loaded or not will return to a stack located in its home stack-cluster located in Yard 2.

The rgv will actually take comparable routes as during adjacent loading. The only difference is that the rgv's will not be handled by their portainers but by on the five stack-cluster warehouse cranes.

**Shuffling from Yard 1 or Yard 3 to Yard 2**

When a feeder vessel has been unloaded and loaded again the containers deposited in Yard 1 or Yard 3 will be shuffled to Yard 2. Like during the adjacent loading of a feeder vessel the shuffling rgv's will take the first inter-berth lane ring.

### 5.5 Number of warehouse cranes

In the chosen layout five warehouses have been placed behind each portainer. As has been stated before, a warehouse hosts two warehouse lanes. Each of these lanes is equipped with a row of three service points for 40' container stacks or six service points for 20' container stacks. An initial assumption has been made that each warehouse hosts one warehouse crane. This brings the total amount of warehouse cranes to 14 * 5 = 70 warehouse cranes. Having assumed this, some initial occupancy rate calculations have been made for three situations.

During the unloading/fast-loading of a container vessel the rgv's assigned to a portainer only deposit and retrieve their containers to and from the stacks located behind the portainer. Given the average portainer production of 1 container per minute the occupancy rate of the five stack-crane has been calculated in Appendix 9. These unloading/fast-loading calculations have been made without interference of rgv's engaged with adjacent loading of other berths as can be seen in Figure 5.11.

![Diagram](image)

*Figure 5.11, portainer production during unloading/fast-loading*

Having the portainers located on Berth 1 unloading/fast-loading a vessel, can increase the occupation rates of the warehouse cranes. By having a vessel adjacent loading on Berth 2, the occupation rates of the warehouse cranes located in Stack 1 will increase. The situation is illustrated in Figure 5.12.
An even busier situation can be created when the warehouse cranes located behind berth 2 not only provide the unloading/fast-loading rgv's of the own portainers but also provide the rgv's arriving from the neighbouring berths 1 and 3, during adjacent loading. These calculations have also been made in Appendix 9.

The results of the heavy occupied and the less heavy occupied situation have been summarised in Table 5.1.

<table>
<thead>
<tr>
<th>Stack location</th>
<th>Stack crane occupancy in stacks behind Berth 1</th>
<th>Microwave crane occupancy in stacks behind Berth 2</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Normal occupancy (Figure 5.7)</td>
<td>Normal occupancy (Figure 5.7)</td>
</tr>
<tr>
<td></td>
<td>Heavy occupancy (Figure 5.8)</td>
<td>Heavy occupancy (Figure 5.9)</td>
</tr>
<tr>
<td>Situation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>20' container stack</td>
<td>33.3 % 58.3 %</td>
<td>33.3 % 77.8 %</td>
</tr>
<tr>
<td>40' container stack</td>
<td>22.2 % 39 %</td>
<td>22.2 % 51.9 %</td>
</tr>
<tr>
<td>Combined</td>
<td>27% 49 %</td>
<td>27% 64.8 %</td>
</tr>
</tbody>
</table>

Table 5.1, Warehouse crane occupancy rates

The warehouse cranes located in the warehouses behind Berth 2 have a maximum occupancy rate of 77.8 % during the unloading/fast-loading of a vessel moored at Berth 2. At the same time rgv's subscribed to portainers located on Berth 1 and Berth 3 are also retrieving containers from the stacks behind Berth 2. This occupancy rate is rather high considering the calculation is an average occupancy rate. With the computer simulation the results of this situation can be analysed more thoroughly. If the capacity of the warehouse cranes will show to be insufficient, warehouse cranes located in the stacks behind Berth 1 and Berth 3 may transfer to Berth 2 in order to increase the stack capacity.
5.6 Number of rgv's on the terminal

The number of used rgv's during a certain procedure will depend on the length of the mean transportation cycle, the portainer production rate and the properties of the rgv, like speed and acceleration. As the transportation cycle length is the most important variable, a calculation has been made for the main transportation cycle length belonging to each handling procedure on the terminal.

The mean transportation distance during the unloading of a vessel is shorter than the mean transportation distance during adjacent loading. In Appendix 10 the mean transportation distances belonging to the different handling strategies on the terminal have been calculated. Based on the mean portainer handling times and the speed of the rgv's on the different lanes the numbers of needed rgv's have been calculated. The results are given in Table 5.2.

<table>
<thead>
<tr>
<th>Procedure</th>
<th>Mean transportation cycle time (seconds)</th>
<th>Number of needed rgv's (rounded off)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unloading a vessel</td>
<td>248</td>
<td>5</td>
</tr>
<tr>
<td>Adjacent loading</td>
<td>529</td>
<td>9</td>
</tr>
<tr>
<td>Shuffling</td>
<td>551</td>
<td>9</td>
</tr>
</tbody>
</table>

*Table 5.2, number of needed rgv's per portainer*

The number of rgv's that will be used during the computer simulation is 1 above the calculated needed number. Rgv's will almost certainly meet each other on crossings resulting in waiting queues on certain nodes. In order to keep the productivity of the portainers high 1 extra rgv might even be the minimum. The computer simulation will start off by using 6 rgv's during unloading and 10 rgv's during adjacent loading and shuffling.
6 Computer simulation

6.1 Problem analysis tool: PROSIM

The designed container terminal will be simulated in PROSIM. PROSIM is a process description method language. When using such a method the model can be observed as a number of components interacting with each other in such a way that they reflect the actual system as close as possible. Building a model means identifying the components and describing their activities in their own module. A computer simulation model will thus consist of different modules, which are connected by the fact that the components interact with each other. These interactions are described in the modules of the components.

A component can either be permanent or temporary. A permanent component will be present in the model permanently whereas a temporary component will only be present for a certain period of time. Every component has certain characteristics that are important for the proper functioning of this component and for the interacting with other components. These characteristics are so called “attributes”. When components have the same type of attributes they form a “class of components”.

A Prosim model consists of two sections:
- A definition section in which all the components and their attributes, sets and other variables are defined. This section is called the DEFINE module;
- A dynamic section in which the dynamic behaviour of the components is described. This section consists of “modules” and “macros”.

A PROSIM model always starts with a module called MAINMOD or Main. This module takes care of the run control and the creation and activation of the first components. The other modules are activated either by the Main module or by another module.

A module can activate a macro. A macro contains a description of certain activities to be executed. These activities are repeated regularly by the module and are therefore put in a macro in order to keep the module as simple as possible.
6.2 Definition of the components of the computer model

The model consists of a number of components interacting with each other in such a way that they reflect the real system as close as possible. The components used in the model are listed below and are explained in the following paragraphs. The total flow of the interacting terminal components is shown in Appendix 11.

<table>
<thead>
<tr>
<th>Components</th>
<th>Role in the model</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;Main&quot;</td>
<td>Takes care of the simulation run &amp; creates lay-out</td>
</tr>
<tr>
<td>&quot;Generator&quot;</td>
<td>Generates the vessels</td>
</tr>
<tr>
<td>&quot;Import generator&quot;</td>
<td>Generates import containers</td>
</tr>
<tr>
<td>&quot;Export generator&quot;</td>
<td>Generates export containers</td>
</tr>
<tr>
<td>&quot;Berthmaster&quot;</td>
<td>Manages the berth distribution</td>
</tr>
<tr>
<td>&quot;Portainermaster&quot;</td>
<td>Instructs the portainers</td>
</tr>
<tr>
<td>&quot;rgv master&quot;</td>
<td>Controls the number of Rgv's at the terminal</td>
</tr>
<tr>
<td>&quot;Trafficmaster&quot;</td>
<td>Creator of rgv routes and instructor of the Rgv's</td>
</tr>
</tbody>
</table>

The following components are "class components":
- Vessel
- Portainer
- Rgv
- Crane
- Stack
- Portloc (portainer location)
- Node
- Section
- Platform

In addition the model uses "macros" which are activated by a module to perform specific actions.
6.2.1 The component Main

Each PROSIM model requires the component Main. The Main takes care of the simulation run by creating components and their attributes. In this case the Main ignites a simulation run by activating the Shipgenerator. The Main also creates the portainers, the cranes, the rgv's, the stacks and the layout of the terminal. Figure 6.1 shows the different tasks of the Main. The shadow behind the first box refers to another process not shown in this figure, the creating of the lay out. This is explained in Paragraph 6.4.

Create portainers
cranes agv's lay-out
stacks and berths

Activate
shipgenerator

Wait 30
days

Cancel
shipgenerator

Terminate all

Figure 6.1, the process of Main
6.2.2 The component ship Generator

The component Generator is responsible for generating the vessels. Before generating a vessel it waits a certain inter-arrival time. This inter-arrival time is defined in the Main and refers to a distribution function. After a vessel is generated the Generator will again wait a certain period of time before creating the next vessel. This will go on forever until the Main stops the Generator. The process of the Generator is shown in Figure 6.2.

![Diagram showing the process of the ship Generator](image)

*Figure 6.2, process of the ship Generator*

6.2.3 Import generator & Export generator

These generators are responsible for creating the containers that will be imported to and exported from the terminal by the vessels. There are two different containers in this model: 20 foot and 40 foot containers. Over a longer period of time the Import generator has to create as many import containers as the Export generator creates export containers. The reason for this is the fact that this is a transhipment terminal and there is no hinterland connection.
6.2.4 The component Berthmaster

The Berthmaster is responsible for assigning the berths to the vessels, and is activated when a vessel is created or when a vessel is leaving the terminal. Once the Berthmaster is activated it looks whether there is a vessel waiting in the anchorage. If there is a vessel waiting in the anchorage the Berthmaster looks whether there is a free berth for this particular vessel. If there is a free berth the Berthmaster will allocate the free berth to this vessel, ask the vessel to moor and inform the Portainermaster of the new vessel at the berth. If there is no free berth the Berthmaster will instruct the vessel to wait in the anchorage.

In the model there are two kinds of berths, one berth for the mega vessels and two for the feeder vessels. If a feeder vessel arrives at the anchorage the Berthmaster first looks whether there is a free feeder berth. If there is only one free feeder berth the feeder vessel will be assigned to that particular berth. When there are two feeder berths available the Berthmaster will assign the feeder vessel to the berth that has not been occupied for the longest time. This way any possible shuffling at the other feeder berth can continue. The mega vessels have just one berth, Berth 2. Once a mega vessel enters the anchorage the Berthmaster looks whether the berth is free, independent of possible shuffling. If the berth is free the mega vessel will be assigned to the berth.

![Flowchart of the Berthmaster process](image)

*Figure 6.3, Process of the Berthmaster*
6.2.5 The component Portainermaster

The model has been designed to contain two feeder berths and one mega vessel berth hosting a total of fourteen locations for portainers. It is possible to handle a mega container vessel with a maximum of six portainers and a feeder vessel with a maximum of four portainers. All the simulation runs in this study have been carried out with the maximum number of 14 portainers at the quay. This is done because the focus of this study lies on the processes on the terminal as opposed to a simulation of the arrival schemes of the container vessels.

It is possible to simulate with less than 14 portainers on the terminal if this is desirable. The main job of the Portainermaster will then be to allocate the portainers to the right location. An option could be to simulate a terminal with only 10 portainers available. This case has not been used in this study but the associated allocation will be discussed below.

Option: One vessel berthed

In case there is only one vessel berthed at the quay the Portainermaster assigns a maximum of 6 portainers to a mega container vessel moored at Berth 2 and a maximum of 4 portainers when it is a feeder vessel. The Portainermaster directs the portainers to the precise portainer locations on the quay. These locations depend on which berth the vessel is moored at.

Case: A mega container vessel is berthed at Berth 2:

![Figure 6.4, mega vessel moored at Berth 2](image)

- The mega vessel moored at Berth 2 gets six portainers assigned on the portainer locations 5 to 10. The four remaining portainers will remain inactive on Berth 1.

Case: A feeder vessel is berthed at Berth 1:

![Figure 6.5, feeder vessel moored at Berth 1](image)
• If moored at Berth 1, portainers 1 to 4 are directed to portainer locations number 1 to 4. The portainers 5 to 10 will remain inactive on Berth 2.

Case: A feeder vessel is berthed at Berth 3:

![Portlocations & Portainers](image)

Figure 6.6, a feeder moored at Berth 3

• If the feeder vessel is moored at Berth 3, portainers 7 to 10 are directed to portainer locations 11 to 14. Portainers 1 to 6 will remain inactive on Berth 2.

Option: Two vessels are berthed

When there are two vessels moored at the quay a number of possibilities can be distinguished. There can be two feeders or a feeder and a mega container vessel.

Case: There are two feeder vessels berthed of which one is moored at Berth 1 and the other is moored at Berth 3.

![Portlocations & Portainers](image)

Figure 6.7, feeder vessels moored at Berth 1 & Berth 3

• The feeder vessel moored at Berth 1 will have the portainers 1 to 4 assigned to it, located on the portainer locations 1 to 4. The feeder berthed at Berth 3 will have the portainers 7 to 10 assigned to it, located on portainer locations 11 to 14.
Case: There is one feeder vessel moored at Berth 1 and one mega vessel moored at Berth 2.

![Figure 6.8, feeder vessel moored at Berth 1 and a mega vessel moored at Berth 2.](image)

If the feeder vessel is moored at Berth 1 it will receive the portainers 1 to 4 located on portainer locations 1 to 4. The mega vessel will receive portainers 5 to 10 located on portainer locations 5 to 10.

Case: There is one feeder vessel moored at Berth 3 and one mega vessel moored at Berth 2.

![Figure 6.9, feeder vessel moored at Berth 3 and a mega vessel moored at Berth 2](image)

- If the feeder vessel is berthed at Berth 3 it will receive portainers 7 to 10 on the portainer locations 11 to 14. The mega vessel will receive portainers 1 to 6 on the portainer locations 6 to 10.

**Option: Three vessels berthed**

When there are three vessels berthed at the quay, the only possibility is that there is a mega vessel moored at Berth 2 and that the two feeder vessels are moored at Berth 1 and Berth 3.

![Figure 6.10, Three vessels moored](image)

The mega container vessel receives six portainers on portainer locations 5 to 10, while the feeder vessels receive 2 portainers each. The portainers on Berth 1 and Berth 3 have variable portainer locations as they handle the entire feeder vessel. This has been done in order to serve the mega vessel as best as possible.

The consequence of working with 10 portainers is the occasional transferring of the portainers to the right portainer locations. The only situation in which the productivity on the berth is
seriously affected is when the terminal is fully occupied. Ultimately there will be an economic optimum for the number of portainers used on the terminal. As in this study the focus will lie on the processes that take place on the terminal, as opposed to a simulation study on vessel arrival schemes, the simulation runs have been carried out with the maximum of 14 portainers.
6.2.6 The component Rgymaster

The Berthmaster allocates a berth to a vessel, and activates the Portainermaster. The Portainermaster in its turn allocates and activates the portainers. Once they have arrived on their allocated location, the portainers will activate the Rgymaster. This is shown in Figure 6.10. Once the Rgymaster is activated the Rgymaster determines the number of active portainers and subsequently, the number of needed rgy's. The number of needed rgy's depends on the assignments of the active portainers at that moment.

If a portainer is unloading a vessel only 6 rgy's per portainer will be needed. If a portainer is fast-loading a vessel, the to be loaded containers will all come from the stack-cluster behind the portainer. In this case also 6 rgy's per portainer will be needed. When a portainer is adjacent loading a vessel, the to be loaded containers will have to come from the stacks located at an adjacent berth. In this case 10 rgy's per portainer will be used. These rgy's are all portainer dedicated. When a portainer is not yet at its right portainer location then no rgy's will be assigned to it.

This situation might occur when the terminal works with less than 14 portainers and when a portainer is not at his allocated location at the moment the Rgymaster is activated. If a portainer has just finished loading a vessel and the vessel is about to leave, the shuffling of the unloaded containers to the correct yard can start.

Once the Rgymaster has determined the needed number of rgy's and the number of already active rgy's he can calculate the difference. If there are not enough rgy's active, the Rgymaster will activate extra rgy's from the parking.

If there are too many rgy's the Rgymaster will give the surplus of rgy's the attribute "byy". This attribute indicates that this rgy has to go to the parking. The next time this rgy asks the Trafficmaster for an assignment, the Trafficmaster will notice the attribute and the next assignment will be to go to the parking. The Trafficmaster will be explained in paragraph 6.2.7. This attribute is necessary because the Rgymaster itself cannot send a rgy to the parking. The reason for this is that the rgy could be in the middle of an assignment, which it has to complete first. The process of the Rgymaster is shown in Figure 6.11.
Figure 6.11, process of the Rgvmaster
6.2.7 The component Trafficmaster

The Trafficmaster is responsible for the assignments of the rgv's and can only be activated by a rgv. Once the Trafficmaster is activated, there are a number of possible assignments for the rgv's but roughly the assignments are divided into two main groups. The first group contains assignments concerning the horizontal transportation between the stack and the berth. The second group assignments concerns the shuffling between the stacks. The Trafficmaster can, by using the attributes of the rgv, determine the state of the rgv, which will lead to the next task of the rgv.

When activated by a rgv the Trafficmaster first looks whether the attribute "bye" of the rgv is true. If it is, the rgv is supposed to go to the parking. The Trafficmaster will remove the rgv from the "active rgv-list", a set containing all the active rgv's, and will activate the rgv from to_parking in the rgv module.

If the rgv is not supposed to go to the parking the Trafficmaster will, based on the state of the rgv's attributes: "yard", "loaded", "y" and "x" of the rgv, decide what the next the next assignment should be. These different assignments require different actions by the Trafficmaster. These actions are all stated in the macros of the Trafficmaster. Figure 6.12 shows the flow chart of the Trafficmaster.

![Flowchart of the Trafficmaster](image)

*Figure 6.12, process of the Trafficmaster*
In Figure 6.12, [1] and [2] refer to processes, depending on the attributes of the rgv. These processes will be explained below.

In case [1] the rgv is assigned to a portainer. The rgv's next assignment from the Trafficmaster depends on the location of the rgv and its attributes. The 6 possibilities are listed below. The rgv can be in the yard next to a service point where it has just delivered or received a container or the rgv is located under a portainer where it has just received or delivered a container.

<table>
<thead>
<tr>
<th>Rgv is in the yard next to a service point</th>
<th>Rgv is at the quay under the portainer</th>
</tr>
</thead>
<tbody>
<tr>
<td>agv_ship is loading</td>
<td>agv_ship is loading</td>
</tr>
<tr>
<td>agv_ship is unloading</td>
<td>agv_ship is unloading</td>
</tr>
<tr>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td>6</td>
</tr>
</tbody>
</table>

*Figure 6.13, possibilities when a rgv is working for a vessel*

The numbers in the matrix refer to the actions of the Trafficmaster.

1. The Trafficmaster will first jump to a macro "loadyardallocation". In this macro, a stack is chosen for the rgv to retrieve a container by using an algorithm. This stack is now an attribute of this rgv. This attribute is called "y". Next the Trafficmaster will jump to the macro "rgv_stack_stack". This macro will create the "route" to this new stack. The Trafficmaster will also put the rgv on the assigned list of the stack. By the assigning of the rgv, the warehouse crane in the stack knows there is rgv on its way and can start retrieving the container to put it on a service point.

2. While the rgv is portainer dedicated the destination of the rgv is clear. The Trafficmaster will only jump to the macro "rgv_stack_berth" to create the route to get from the stack to the portainer location.

3. The Trafficmaster will jump to the macro "rgv_unload_back". This macro creates the route to get back from the yard to the portainer. The Trafficmaster will also assign the rgv to the portainer. When the rgv is assigned the portainer knows the rgv is coming and can start retrieving a container from the vessel.

4. The Trafficmaster will jump to the macro "loadyardallocation" and then it will jump to the macro "rgv_berth_stack". "Loadyardallocation" chooses the stack and "rgv_berth_stack" creates the route to get to this stack. The Trafficmaster will also put the rgv on the assigned list of the stack. By the assigning of the rgv, the warehouse crane in the stack knows there is rgv on its way and can start retrieving the container to put it on a service point.

5. This situation does not occur too often. It only occurs when a rgv has received the last import-container from a vessel and asks the Trafficmaster for an assignment. In this case the Trafficmaster will jump to the macro "unloadyardallocation". Here an algorithm decides where the rgv has to deliver the container. This stack is now an attribute of the rgv. Next the Trafficmaster jumps to the macro "rgvUnload", where the "route" of the rgv is created.
6. The Trafficmaster will jump to the macro "unloadyardallocation". In this macro the stack is chosen where the rgv has to deliver the unloaded container. Next the Trafficmaster will jump to the macro "rgv_unload", where the "route" to the stack is created.

In case [2], the rgv is not attached to a portainer handling a vessel but is shuffling between the stacks located in the different yards on the terminal. During the shuffling process a rgv has been detached from its portainer. Instead of returning to its former portainer, it is now subscribed to the stack-cluster located behind the portainer. The shuffle containers, retrieved from stack cluster, will be delivered to one of the stacks located in the adjacent yard. If there is a container to be shuffled back the rgv retrieves it and brings it back one of the stacks located in its stack-cluster.

There are two states in which a shuffling rgv can activate the Trafficmaster in its shuffling section. Below these 2 possibilities are explained in succession.

1. *The rgv is not loaded and is located under the portainer for the last time.*
   This is the case when an empty rgv has just delivered its last container to the portainer loading a vessel. Every rgv experiences this state only once during the handling of a vessel. In this case the Trafficmaster will first jump to the macro "yardshuffleallocation". In this macro a stack, located in the stack-cluster behind its former portainer, is found. From this stack the rgv will retrieve a shuffle container. This stack becomes an attribute of this rgv. Next, the Trafficmaster will jump to the macro "rgv_berth_stack". This macro creates the route from the berth to the stack.

   The Trafficmaster will also assign the rgv to the stack. By the assigning of the rgv, the warehouse crane in the stack knows there is rgv on its way and can start retrieving the container to put it on a service point.

2. *The rgv has just received a shuffle container and has to deliver it to another stack.*
   The Trafficmaster will first jump to the macro "yardsuffleallocation". In this macro the stack, located at an adjacent yard, is chosen. The rgv has to deliver the container to this new stack. Next the Trafficmaster will jump to the macro "rgv_stack_stack". In this macro the "route" to the new stack is created.

   The Trafficmaster will also assign the rgv to the stack. By the assigning of the rgv, the warehouse crane in the stack knows there is rgv on its way. If there is a shuffle container in the new stack destined to be taken away, the warehouse crane can start retrieving the container and put it on a service point.

Another important task of the Trafficmaster is keeping count of the number of containers that have to be loaded, unloaded and shuffled. Each vessel that arrives at the berth has an attribute "total unload", referring to the number of containers to be unloaded. Whenever the Trafficmaster gives a rgv an assignment to get a container from the vessel it deducts one from the "total unload". When the "total unload" of a vessel becomes zero, the Trafficmaster removes the vessel from the set "unloadingvessel" and joins it to the set "loadingvessel".

When the vessel is being loaded the Trafficmaster also keeps count of the containers. The macro "container-yeardallocation" is used for this. The number of loaded containers is called "tmtotal". Tm stands for Trafficmaster. When "tmtotal" is zero, the vessel is fully loaded again and ready to leave the terminal.
6.2.8 The class component Vessel

There are two kinds of vessels, the mega container vessels and the smaller feeder vessels. After a vessel has been generated, it will activate the Import & Export generator that will create the containers for the vessel. The vessel will then enter the anchorage and activate the Berthmaster, while waiting for permission to berth. Once permission is granted the vessel will berth and wait while it is unloaded and loaded again. When the vessel is loaded it will leave the berth and activate the Berthmaster. After it has left the berth the attributes of this vessel will be recorded and the vessel will be terminated.

Figure 6.14, process of the class component vessel
6.2.9 The class component Portainer

Portainers are designed for the transport of containers between the quay and the vessels. The Portainer master assigns the portainers to the vessels and sends them to the right locations on the quay. These portainer locations in the simulation program are called “portlocs”. In the simulation runs 14 portainers are used for 14 portainer locations. This means the portainers will always be on the right location. It is possible to work with less portainers in the program.

The portainer can be activated by the Portainer master or by an assigned rgv. If it is activated it checks whether it is located at the right portainer location. If not, it will try to get to the right location this only happens when less than 14 portainers are used. It may happen that a portainer, that is not at its the right location on the quay, has to wait before it is allowed to move because other portainers that are still working, are standing in the way. These portainers are supposed to clear the way as soon as possible (such portainers may be in the process of unloading or loading which has to be finished first).

Once the portainer has arrived at the correct location it will wait for a rgv to be joined to the assigned list of the portainer. If this assigned rgv is an empty rgv the portainer will get a container from the vessel and wait for the rgv to arrive under the portainer, in the portainer queue. When the rgv arrives, the portainer will put the container on the rgv. Once the unloading of the vessel is finished, the portainer will start the loading of the vessel at the arrival of the first loaded rgv under the portainer.

![Flowchart](image)

*Figure 6.15, process of the class component portainer*
The grey area in the flow chart in Figure 6.15 indicates this box is part of another not shown process, it is the process where the portainer transfers to its right location.

Figure 6.16, flow chart of the portainer while it is going to the right position at the quay
6.2.10 The class component Rgv

Rgv's are used for the horizontal transportation between the quay and the stack. One rgv is assumed to carry a maximum of one container; this can either be a twenty-foot or a forty-foot container. The Rgvmaster activates the rgv's. Once a vessel is moored at the quay and the portainers are activated the Rgvmaster decides, using the number of active portainers, how many rgv's to retrieve from the parking. The Rgvmaster joins all the needed rgv's to an "active rgv" list and to an "rgv worklist". The "active rgv list" is a set containing all the active rgv's. The rgv worklist is a set containing all the rgv's requesting the Trafficmaster for work. Once joined to the rgv worklist, the rgv activates the Trafficmaster. The Trafficmaster removes the rgv from the rgv worklist, decides where the rgv has to go and activates the rgv in the rgv module. The flow chart concerning the Trafficmaster shows that there are some different possible options from where to activate the rgv.

Option: The Trafficmaster activates the rgv from "toyard" in the rgv module

Toyard means that the rgv is not in the yard but at the quay and is destined to go to the yard. This as opposed to "inyard" where the word "in" indicates that the rgv is already in the yard and has to go to another place in the yard, for example during shuffling. In this case the rgv is activated in the rgv module from toyard, so the rgv jumps to the macro called "rgv_berth_stack". When the rgv enters the macro its present and the future location on the terminal are known. The macro creates the route, which the rgv has to follow in order to get to the future location, using algorithms. In this case the route is from a berth to a stack. The rgv drives using its route, to the right location and enters the warehouse crane queue to and waits to be serviced.

Option: The Trafficmaster activates the rgv from "inyard" in the rgv module.

If the rgv will be activated in the rgv module from "inyard", the rgv jumps to a macro called "rgv_stack_stack" in the Trafficmaster. When the rgv enters the macro its present and future location at the terminal are known. The macro creates the route, which the rgv has to follow in order to get to the future location, using algorithms. In this case the route is from a stack to a stack. The rgv drives using its route, to the right location, enters the crane queue and waits to be serviced.

Other options are that the Trafficmaster activates the rgv from "toquay", "inquay" and "toportainer". The macro's respectively jumped to by the rgv to receive the routes are "rgv_stack_berth", "rgv_unload_back", "rgv_unload" and "Rgv_toparking".
Figure 6.17, process of the rgv

The grey parts in Figure 6.17 contain the driving processes of the rgv. This driving process is shown in chapter 6.5.
6.2.11 The class component Crane

There is one warehouse crane on every pair of stacks. A pair of stacks contains two stacks, two warehouse lanes and two rows of service points. The warehouse crane is situated in between these two service point rows, as can be seen in Chapter 5. The warehouse crane is responsible for the horizontal and vertical transport from the service points to the racks of the warehouse and back. The warehouse crane will not be active until a rgv is assigned to the warehouse crane or until a rgv has delivered a container onto a service point.

The Trafficmaster can assign a rgv to a warehouse crane. When for example the Trafficmaster has just ordered an empty rgv to collect a container from a stack, the Trafficmaster will join the rgv to the assigned list of the warehouse crane working on that stack. The moment the rgv is on the assigned list the crane becomes active. The warehouse crane checks the list and learns there is a rgv coming with a container request. The crane starts to retrieve the container from the stack and loads it onto a service point. When subsequently the rgv arrives in the stack the container is already on the service point and ready to be taken away.

Another possibility is that a rgv arrives in a stack to deliver a container. In this case the rgv is not on the assigned list of the warehouse crane, because the warehouse crane cannot work in advance. When this rgv arrives at the stack a service point shuttle will put its container on a service point. The warehouse crane will notice there is a container on the service point and will deliver it to the right place in the stack.

![Diagram](image)

*Figure 6.18, process of the class component crane*
6.3 The macros used in the program

In addition to the modules the program uses macros. Macros are used to perform specific actions. The different macros used will be explained here. The listings of the macros are shown in Appendix 28, the listing of the program.

- Create platform, this macro is responsible for creating the platforms and is used by the Main. This is explained in paragraph 6.4.
- Station, this is a sidetrack and is used by a rgv when it leaves the main track in order to be served by a portainer.
- Short cut, this is a sidetrack used when a rgv wants to leave the main track to take a shortcut, or enter a stack.
- Stacking crane is a macro used by a rgv leaving the main track in order to enter a warehouse lane to be served by a warehouse crane.
- Rgv_to_parking is a macro used by the rgv in order to get to the parking
- Rgv_unload is a macro that creates the route for a rgv to get from the berth to a stack. This macro is used by the Trafficmaster to determine the route for the rgv when the rgv is unloading a vessel.
- Rgv_unload_back is a macro used by the Trafficmaster to create the route for a rgv when unloading a vessel. This route is from a stack to a berth.
- Rgv_stack_berth is a macro used by the Trafficmaster that creates a route from any stack to any berth.
- Rgv_berth_stack is a macro used by the Trafficmaster to create a route from any berth to any stack.
- Rgv_stack_stack is a macro the Trafficmaster uses to create the routes between the stacks.
- Create_stack is used by the Main when it creates the layout of the terminal. In this macro the stacks are created.
- Super_shortcut is a sidetrack used by a rgv when it wants to enter a shortcut at a place at the terminal connected to two or more sidetracks.
- Unloadyardallocation is a macro that determines the stack locations for the rgv's to deliver their containers to.
- Loadyardallocation is a macro that chooses where the rgv has to get a load container.
- Conversation is the macro that creates the control box.
- Preclaimnodes is a macro called upon by the rgv responsible for claiming nodes when more than one node needs to be claimed.
- Rgv_uneasy_b_s this is a macro that creates a route from a berth to a stack in the transition between loading a vessel and shuffling at a portainerlocation.
- Yardshufflleallocation is the macro that creates the locations for the rgv's to get containers from when shuffling.
- Y_creation is a macro responsible for determining which stacking crane works in which stack.
- Create_outportation is the macro responsible for creating outportations.
- Crane_cycletime is a triangular function that can be used when creating crane cycle times.
- Crane_outportations is the macro that creates output files for all the cranes used at the terminal. With these files the mean crancycletimes can be calculated.
- Creating sections is a macro that attaches existing queues of sections to windows in order to monitor these queues. In this way node rows and stack rows can be monitored.
- Create_portout is a macro that creates portainer outportations in order to monitor the queues at the portainers.
- Portainer times is a macro responsible for creating the portainer cycletimes here four functions are used. Two for the loading and unloading of the mega vessels. And two for the loading and unloading of the feeder vessels.
6.4 The layout of the terminal as created in the Main

Appendix 2 shows the layout of the terminal as it is created in the Main. To understand the creation of the terminal layout some used components must be explained. These are the class components node, section, stack, port location and platform.

The class component node

The structure of the terminal consists of nodes. Nodes are points in the 2-dimensional layout area of the terminal. By giving these nodes different attributes driving lanes and sections can be created. Important attributes of a node are: claimed, node speed, side_track and outs[1], outs[2] and outs[3]. These attributes will be explained later on the basis of some examples. The 800 nodes and their attributes are created in the Main module.

The class component section

This class component section is a so-called dead class component. Dead class components differ from other class components in the fact that they do not have their own module in the simulation program. These components do not do anything in the system but the other components can do something with them. Dead class components are only used to carry information.

Sections can be one way driving lanes created between nodes. These one way driving lanes are called the "main tracks" of the terminal. These main tracks are created in the Main through an input file. In Figure 6.19 a part of this file is given as an example; the complete input file is shown in Appendix 16.

When the Main creates the layout of the terminal, it first reads the first character string of the input file. In this case the number "43". This number refers to a node, node[43]. The second and third character strings are the x and y co-ordinates of this node, "8" and "178". After reading a second node and its attributes the main creates a section between node[43] and node[44]. This section is called section 43. An important attribute of this section is its length called "s_length". The function "distance_to" calculates this s_length. After creating section 43 node[43] receives a new attribute "outs[1]". Outs[1] refers to section 43. The endpoint of section 43 is node[44].

Figure 6.19, the creation of a main track by the Main
The class component stack

Stacks are created in the main using a macro and an input file. Some important attributes of the stacks are a stackrow, a stacknumber and a way in and out of the stacks. Whenever a rgv enters a stack it can do so by driving to the stack_in_node. This is a reference to a node. Via the way_in_stack, a reference to a section, the rgv enters the stack.

By entering the stackrow the rgv can be served. In the end the rgv will leave the stack using the way_out_stack. In Figure 6.20 the file for the creation of stacks is given.

<table>
<thead>
<tr>
<th>warehouse lanes</th>
<th>stack21</th>
<th>231</th>
<th>260</th>
<th>183</th>
<th>148</th>
</tr>
</thead>
<tbody>
<tr>
<td>stack22</td>
<td>259</td>
<td>232</td>
<td>183</td>
<td>134</td>
<td></td>
</tr>
<tr>
<td>stack23</td>
<td>233</td>
<td>256</td>
<td>183</td>
<td>131</td>
<td></td>
</tr>
<tr>
<td>stack24</td>
<td>257</td>
<td>234</td>
<td>183</td>
<td>119</td>
<td></td>
</tr>
<tr>
<td>stack26</td>
<td>235</td>
<td>256</td>
<td>183</td>
<td>116</td>
<td></td>
</tr>
</tbody>
</table>

Figure 6.20, input file for creating warehouse lanes

The Main reads the first character string from the file. This is the name of the stack, "stack21". The second character string is the entrance node of the stack. In this case node[231]. The third character refers to the endpoint of the stack, node[260]. The fourth and the fifth character strings refer to the actual x and y co-ordinates of the middle of the stack. The main creates the stack by creating 2 sections a section "way-in" and a section "way-out" of the stack.

![Diagram of warehouse layout]

Figure 6.21, the creation of a stack

After creating the stack, the entrance node of the stack, in the case of "stack21" node[231], will receive an attribute outs[2]. Outs[2] is a reference to the section "way-in". Figure 6.21 shows the creation of the stack.

A rgv can thus enter stack21 by leaving the main route at node[231]. It subsequently enters the section way-in. The warehouse lanes are the blue lanes. The arrows point out the driving directions.
Shortcuts

Shortcuts are lanes that can be created between two nodes that are not connected by a main track. Shortcuts can be used as a connection between two main routes. A rgv can enter a shortcut like it enters a stack, via a node. Shortcuts are created using an input file. An example of a part of this input file is given in Figure 6.22.

![Figure 6.22, input file for creating shortcuts](image)

The Main reads the first character string, in this case 38. This refers to node[38], the entrance of the short cut. The second character string of a node refers to the end point of the shortcut. The shortcut itself is the section between these nodes. After the main has created the shortcut the entrance node of the short cut will receive an attribute outs[2]. An example of a shortcut is given in Figure 6.23.

![Figure 6.23, the creation of a shortcut](image)

The shortcuts are the green lanes between the red main tracks.

The class component portloc

The class component portloc is created in the Main. A portloc is a place where a portainer can be situated, a portainer location. There can be more portlocs than portainers so it is possible for a portloc to be unoccupied. This can be the case when a simulation is carried out with less than 14 portainers.

Every portloc has a set called "portlocqueue". This set contains the rgv's waiting at that portlocation to be served. The lengths of the portlocqueues during a simulation are recorded and can be watched during or after the run.

Another attribute of a portloc is the set "assigned rgv's."
This set contains rgv's performing a task for the portainer located at that portloc. When a rgv is assigned, the portainer knows it cannot move to another portloc. This is to avoid a rgv driving to an unoccupied portloc.

**The class component Platform**

This class component platform is created in the Main by jumping to the macro "create_portainerlocation". A portainerlocation consists of two platforms. This has everything to do with the layout of the terminal. These platforms are also created via an input file. An example of this file input is given in Figure 6.24.

<table>
<thead>
<tr>
<th>@portainers@</th>
<th>loc3los</th>
<th>loc3laad</th>
<th>loc3loc</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>242</td>
<td>249</td>
<td>163</td>
</tr>
<tr>
<td></td>
<td>248</td>
<td>243</td>
<td>163</td>
</tr>
<tr>
<td></td>
<td>563</td>
<td>563</td>
<td>563</td>
</tr>
</tbody>
</table>

*Figure 6.24, input file for creating a platform*

The Main reads the first character string, in this case "loc3los". This is the name of the platform. The second and the third character strings refer to the entrance and the exit node. The fourth and the fifth character strings refer to the x and y co-ordinates of the actual location of the platform. In Figure 6.25 the creation of the platform is given.

*Figure 6.25, the creation of 2 platforms and a portainer location*

The blue area in Figure 6.25 is called a portainer location. The black lines are the platforms. Each portainer location has two platforms. Meaning it has two ways for a rgv to enter the portainer location. In the end they will both end up in the same "portlocqueue". Loc3los and Loc3laad are the names of the platform.
6.5 The traffic rules at the terminal

6.5.1 The priority rules

The layout of the terminal is divided in main tracks, short cuts, platforms and stacks as has been explained in the previous section. Once a rgv is on a main track, the red lanes in Figures 6.19, 6.21, 6.23 and 6.25, it has priority over rgv’s entering from a sidetrack. Sidetracks can be short cuts, platforms or stacks.

When driving on the main track the rgv automatically claims the next node that enters the claiming zone of the vehicle and declaims this node when it has past the node. Another rgv can only enter the main track when the entrance node onto the main track is not claimed. So when a rgv has been served at a stack and wants to enter the main track it has to wait until the entrance point, the endpoint of the stack, is not claimed. This implies that a rgv on the main track only stops when it has to leave the main track.

Another rgv driving behind this rgv will always leave enough space for the first rgv to brake, to turn its wheels and leave the main track. This space is called the claiming zone of a rgv. As a consequence of the priority rule, there will be no queuing on the main track, but it is possible a rgv has to wait when it wants to enter the main track coming from a side track. This is shown in Figure 6.26.

At all the entrance points of the main tracks room must therefore be reserved for the waiting rgv’s that are not yet permitted to enter the main track. This waiting area is available in the green section.

![Diagram of traffic rules]

In Figure 6.26 Rgv 1 is driving on the main track. During this driving on the main track Rgv 1 automatically claims the nodes within claiming distance. Once a node is claimed its attribute “claimed” becomes 1. Rgv 2 coming from a shortcut wants to enter the main track at node 44 but has to wait until “claimed” of node 44 becomes 0 again. “Claimed” of node 44 can only become 0 again when Rgv 1 has passed the node and changes the attribute “claimed” of the node to 0.

While Rgv 2 is not permitted to enter the main track it has to wait in the queue before node 44. The lengths of these queues are recorded by the simulation program and can be analysed during or after the simulation run. The green sections also enable a rgv to leave the main track without hindering the traffic at the main track.
An enlargement of the area around node 44 is given in Figure 6.27.

![Figure 6.27, space for waiting queue at node 44](image)

### 6.5.2 The driving of a rgv

A rgv drives over the terminal using its own attributes and the attributes of the nodes on the terminal. Two of the most important attributes of the rgv are the set "route" and the set "entrance". The route of the rgv is a set containing all the required exit nodes from a main track. Entrance is a set of the rgv containing only the starting point of the journey. This is a reference to a node. The Trafficmaster creates these sets before the rgv starts its journey. The most important attributes of the nodes are the "outs[1]", the "outs[2]", the "outs[3]" and the "claimed" of the nodes.

The rgv starts its journey at its entrance node. This can be the exit of a stack or the exit of a portainer location. When the rgv starts the entrance node becomes the current node although the rgv has not yet entered the main route. The rgv can only enter the main route when the current node is not claimed. If the current node is not claimed, the rgv enters the main track and claims the current node and starts driving.

Once the rgv enters the main track the rgv will automatically follow the main track driving direction and claim all the nodes within claiming distance. This process continues until the rgv arrives at the first node of its set "route". This node is called the exit node. When the rgv arrives at the exit node the rgv will leave the main track and enter a short cut, a portainer queue or a stack queue.

If the rgv enters a stack or a portainer the rgv will enter the waiting queue and will wait to be served. If the rgv enters a shortcut it will pass the shortcut and will try to re-enter the next main track. This process continues until the rgv reaches its destination. The driving process of the rgv is shown in Figure 6.28. This process always starts at a portainer location or in a stack.
Figure 6.28, flow chart of the driving process of a rgv
In Figure 6.29 the flow chart process of the rgv is applied to a situation on the terminal. In this example the rgv has to go from a portainer location to stack 28. The set "route" of the rgv contains just one node, node 253. The set "entrance" contains one node, node 249.

1. The entrance node, node 249 becomes the current node of the rgv. The rgv has to wait while the current node is claimed by another rgv.
2. When current node is not claimed anymore the rgv is permitted to enter the main track.
3. The rgv enters the main track and claims all the nodes within claiming distance.
4. The rgv starts driving on the main track, automatically claiming all the nodes within the claiming distance of the rgv. By claiming node 250 and 251 the rgv makes it impossible for rgv's coming from stack 29 or stack 40 to enter the main track.
5. After the rgv has passed node 250 and 251 it declares these nodes. The rgv will continue driving until it reaches node 253, the exit node. At this node the rgv will leave the main track and enter stack 28.
7 Verification of the simulation program

Normally when a simulation program has been written the process of checking whether the program is correct, consists of two phases. The first phase is the verification process in which is checked if the simulation program acts like it is supposed to do. The second phase is the validation phase where the simulation program is compared to results gained by experience. In this study only a verification study has been made. The main reason is that there is no experience with this type of terminal, which is necessary for the validation process.

Before any reliable results can be obtained from simulation runs, the program must be checked for inaccuracies and errors that may influence the results. Inaccuracies might be present in the computer simulation language or in the design of the program. The compiler easily traces inaccuracies in the computer language. The compiler is a function that indicates all the errors in the different modules and macros.

Inaccuracies in the program itself are more difficult to trace. They can be traced in two different ways; by stepping through the program and checking the state analyses of the components or by simulating with different extreme values for the input parameters and observe whether the results are as expected.

In the first paragraph the algorithms that determine the destinations and the routes of travelling rgv's have been inspected. Next, in the second paragraph, the travelling rgv itself will be followed closely. In Paragraph 8.3 the consequences of a blocked node are illustrated. The influence of the increase of the number of rgv's per portainer on the rgv cycle times are dealt with in Paragraph 8.4, followed by an impression of the portainer handling function. Finally the influence of the number of rgv's on the ship dwell times at the berth has been verified.
7.1 Verification of algorithms

During a simulation rgv's are send to different areas on the terminal depending on their tasks. To determine the destination of a rgv, the program uses so called allocation macros. If, for instance, a rgv has just received a container from a portainer it will call on an allocation macro in order to receive a stack number.

Since the rgv's are always dedicated to one particular portainer it is not necessary to receive a portainer number after each assignment. The portainers can be seen as a base from which the agv's operate from during the unloading or the loading of a vessel. As a result the allocation algorithms only provide the agv's with a stack number.

After the allocation macros have determined the destination of the rgv, the rgv will also receive a route in order to get there. This route is a collection of exit nodes. When the rgv arrives at such an exit node the program identifies the node as an exit and instructs the rgv to leave the main track it is on. The macros that provide these points are the route macros.

7.1.1 Verification of the allocation algorithms

In the allocation macros "unload-yard-allocation", "load-yard-allocation" and "yard-shuffle-allocation" rgv's receive their new destinations in the yard. During the unloading of a vessel, a just loaded rgv, located under its portainer, will call upon the "unload-yard-allocation". This macro will use its algorithms to determine where the rgv has to deliver the newly acquired container.

After the vessel is unloaded the rgv's will retrieve the containers from the stack so that the vessel can be loaded. Once a rgv has delivered a container to its portainer, it will call upon the loadyardallocation in order to receive a stack number from where it will retrieve the next container.

The rgv's will start shuffling when the last container has been loaded. During shuffling the rgv calls upon the yardshuffleallocation every time it needs a new destination in the stacks. It is important to know that the algorithms used in the allocation macros are correct, in the way that the distributed allocations are random and evenly divided.

The three allocation algorithms have been monitored by attaching outstrings to their results. These outstrings contain the "Y" values. A "y" value represents the destination of the rgv. If, for instance, y is 1 it means the destination of the rgv is warehouse lane 1, located in Yard 1. These values are registered and converted to an output file. In this file the stacks or warehouse lanes and the number of rgv entries have been printed. This has been plotted in a histogram as can be seen in Figure 7.1. If one stack is visited considerably less or more than other stacks this can be easily spotted.
Allocation of stacks during unloading/fast-loading of a vessel

Rgv's unloading a vessel deliver the containers to the available stacks located behind the portainer they are working for. The rgv receives the stack number of the destination stack from the macro "unloadyardallocation". This unloading run has been performed with a mega vessel moored at Berth 2. Six portainers have been used in the unloading process.

![Unload yard allocation](image)

*Figure 7.1, allocation of stacks in Yard 2 during the unloading of a mega vessel*

Figure 7.1 shows a relative higher concentration of rgv entries in the stacks containing the 20' containers, these are the stacks 61-64, 71-74, ..., 111-114. This trend is the result of the equal amount of 20' and 40' containers in the yard due to the assumed TEU factor of 1.5. Because the ratio 20' stacks to 40' stacks is 2:3; the ratio of rgv entries in the 20' stacks and 40' stacks also equals 2:3.

Another important issue is that during the unloading of a vessel half of the warehouse lanes available behind the portainer are used. This is caused by the orientation of the warehouse lanes in the stacks. Warehouse lanes 61 to 70 are located behind portainer 5, but only the even warehouse lanes are used. These are the ones orientated in the direction of the unloading cycle. This does not mean that the stacks located on top of the uneven warehouse-lanes are not used while unloading. The warehouse crane is located between an odd and an even numbered warehouse lane and can deliver or retrieve a container from all the racks. As explained in Chapter 4, Figure 4.12.
Allocation of stacks during the adjacent loading of a vessel

Figure 7.2 is the result of a run with a single feeder vessel moored at Berth 1 during adjacent loading. The consequence is that the stacks located in Yard 2 behind Berth 2 supply the containers for the feeder vessel moored at Berth 1. As can be seen in Figure 7.2, these are the stacks 61 to 120. The stacks located behind Berth 1 and Berth 3 are not involved in the loading procedure.

Figure 7.2, allocation of the stacks during the adjacent loading of a feeder vessel moored at Berth 1

As is the case during unloading, the 20' container stacks have a higher average concentration of agv entries. During adjacent loading all the stacks are involved as opposed to the unloading/fast-loading of a vessel.
Allocation of stacks during shuffling

For the verification of the "yard-shuffle-allocation", a shuffle procedure has been simulated. After a feeder vessel, moored at Berth 1, has been unloaded and loaded, the containers unloaded in Yard 1 will be transported to Yard 2. During a shuffle cycle a rgv makes two allocation requests. One to retrieve a shuffle container from a stack located in Yard 1 and one to deliver the concerning container to a stack located in Yard 2. The stacks from which the containers are to be retrieved are the stacks 1-60. The destination stacks are the stacks 61-120 located behind Berth 2 in Yard 2. The stacks 121-180 in Yard 3 behind Berth 3 are not involved in this run.

![Diagram](https://via.placeholder.com/150)

**Figure 7.3, rgv entries in the stacks during a shuffle process**

The stacks holding the 20' containers are, again, visited more frequently, which complies with the expectations.

7.1.2 Verification of route generating algorithms

The macro's agv-stack-stack, agv-berth-stack, agv-stack-berth, agv-unloading and agv-unloading-back are used to create the routes of the rgv's to the desired locations like allocated stacks or portainers. As was stated in the introduction of this chapter, the routes of the rgv's consist of exit points. If a rgv arrives at such a point it will exit the track it is on. The route generating algorithms always generates the exit nodes that lead to the shortest route.

Possible errors in the algorithms can be traced by performing simulation runs were all the different route possibilities are created. When a rgv has an error in its route it means the rgv will never arrive at the allocated location. Instead it will generate an error or the rgv will be caught in a loop. In both cases the errors are easily observed.
7.2 Verification of the route of a travelling rgv

It is essential for the simulation that the rgv's move exactly according to the planned route and planned time scheme. By following all the movements a rgv carries out during the transportation of a container, the various stages that take place have been checked. Only one rgv has been activated in order to keep the tracing simple. Acceleration, wheel turning and the loading and unloading of containers by the warehouse cranes and the portainers have been checked. An explanation of the various driving situations that take place and the step by step trace can be found in Appendix 12.
7.3 Interaction between different rgv’s at crossing nodes

When a rgv wants to enter a main track at a node that is already claimed by another rgv, it will have to wait until the latter has passed by and clears its claim on the node. To check whether a rgv is waiting while the node is blocked, some nodes at shortcuts have been artificially claimed during a run. The expected result is a jam behind the shortcut where the rgv’s will wait in vain to enter a main track.

As an example Node 504 has been artificially blocked. Even rgv’s have to cross Node 504 when they are providing the Portainers 1 and 2 with containers from Yard 2 during adjacent loading.

![Graph of queue behind claimed node](image)

Figure 7.4, the queue behind a claimed node

In Figure 9.4 the horizontal axis projects the time, which is registered in seconds. The number of rgv's in the queue is registered on the vertical axis. After 986 seconds Node 504 was artificially blocked. This automatically prevents all the rgv’s from crossing the node. A queue of waiting rgv’s is the result.
7.4 A qualitative analysis of the rgv cycle times

Having verified the behaviour of the rgv on individual rgv level in Paragraph 7.2, it is also essential to analyse the rgv cycles for each of the 3 possible rgv cycles. These are the rgv cycles during the unloading of a vessel, the loading of a vessel and when shuffling containers from one stack to another. Increasing the number of rgv's per portainer will influence the rgv cycles.

By plotting the duration of the rgv cycles a qualitative interpretation can be made. The average time needed for a rgv to complete each of these cycles is the average rgv cycle time. In the following paragraphs each of the 3 rgv cycles will be further described and analysed.

The rgv cycle times during the unloading of a vessel.

During the unloading of a vessel the rgv cycle starts under the portainer where the rgv receives a container. The rgv will deliver the container to one of the stacks located behind the unloading portainer. After the delivery the rgv returns to the portainer to receive the next container.

A rgv will unload its containers in the stacks located directly behind the portainer. This implies that an unloading rgv will be influenced during the transportation by rgv's that work for the same portainer and by rgv's that work for an adjacent active portainer. Using one active portainer with one rgv during the unloading of the vessel will eliminate these effects. This way an undisturbed average rgv-cycle time per portainer location can be determined.

The relative longer travelling distances to the stacks behind the even portainers cause the fluctuation in the series with one rgv, which can be seen in Figure 7.6. This as opposed to the shorter travelling distances to the stacks behind uneven portainer locations. The undisturbed average cycle time of one rgv on an uneven portainer location is 3.9 minutes while a rgv located on an even berth has an undisturbed average cycle time of 4.5 minutes.

If the numbers of rgv's per portainer are increased the expectation is that the average cycle times of the rgv's will also increase. More rgv's per portainer equals a higher chance of rgv's interfering with each other on crossings, resulting in waiting times.

The runs with more than one rgv have been carried out with 14 active portainers on the three berths, causing interactions between rgv's of adjacent portainers. If the number of rgv's is increased to 2 or even to 6 per portainer, the average rgv cycle time will indeed be longer as
can be seen in Figure 7.6. With 6 active rgv's the average cycle time for even portainers is about 5 minutes, while the average cycle time for uneven portainer is about 5.5 minutes.

![Rgv cycle times when unloading a vessel](image)

**Figure 7.6, rgv unloading cycle times when unloading a vessel**

The rgv cycle times during the adjacent loading of a vessel.

During the adjacent loading of a vessel the rgv cycle starts in the stack, where the rgv receives a container. The agv will deliver the container to its portainer located on one of the three berths. After the delivery the rgv returns to a stack to receive the next container.

![Diagram](image)

**Figure 7.7, loading cycle of an rgv**

A vessel can be loaded in two different ways. One method is by loading containers that have been shuffled to the correct stacks located behind the loading portainer. The routes of the loading rgv's are similar to routes unloading rgv's cover. Verification has demonstrated that the rgv cycle times during fast-loading are the same as during unloading. Therefore Figure 7.6 can also be interpreted for fast-loading.

When there has been insufficient time on the terminal to shuffle all the containers to the correct positions behind the loading portainers, the rgv's will have to retrieve the containers from an adjacent berth, as explained in Chapter 3.5. This means the rgv will have to cover a greater distance in order to collect a container resulting in a longer rgv cycle time.
During 'adjacent' loading the rgv's of the portainers on Berth 1 will retrieve their containers from the stacks situated in Yard 2 behind Berth 2. As can be seen in Figure 7.8, the rgv cycle times of the rgv's subscribed to the portainers on Berth 1 decrease from Portainer 1 to Portainer 4. This is plausible, as Portainer 1 is located further from Yard 2 than Portainer 4.

Rgv's subscribed to the portainers on Berth 3 will also retrieve their containers from Yard 2 resulting in the same rgv cycle times as the rgv cycle times of the rgv's subscribed to the portainers on Berth 1. Berth 2 is the exception because the rgv's will either retrieve their containers from Yard 1 or Yard 3 resulting in fairly invariable rgv cycle times.

This basic verification run has not been carried out with 1 agv per portainer but with 2 rgv's per portainer, because the rgv flow during adjacent loading is separated into two directions. Even rgv's take the bottom inter-berth lanes located on the berths whereas uneven rgv's take the top inter-berth lanes located behind the yards. A second run with 12 rgv's illustrates that by increasing the number of rgv's the rgv cycle times will be longer. This is plausible since more rgv's amount to more interactions between rgv's.

![Adjacent loading cycle](image)

*Figure 7.8, rgv cycle times when adjacent loading a vessel.*

The **rgv cycle times for shuffling on the terminal**.

A shuffle cycle starts at the supplier of containers. In this case the supplier is one of the 10 stacks located behind the portainer the rgv was subscribed to during the unloading and the loading of the vessel. The rgv will always return to this cluster of stacks to retrieve a shuffle container. Stack-cluster 1 for instance is the 10 stacks located behind Portainer 1.

Once the rgv has received the container, it will head for a stack located behind the berth of destination. Rgv's with origin Berth 1 and Berth 3 will shuffle containers from Yard 1 and Yard 3 to Yard 2. Rgv's with origin Berth 2 will shuffle containers from Yard 2 to Yard 1 and Yard 3. After delivering the container there are two possibilities, depending on the fact whether there are containers to be shuffled back. Either the rgv will receive a new shuffle container and deliver it to the rgv's original stack-cluster, or the rgv will return empty to its original stack-cluster.
When comparing Figure 7.8 and Figure 7.9 the similarity is noticeable. Like Figure 7.9 the two most remotely located stack-clusters have the longest rgv cycles. The rgv cycle times of the rgv's operating for the stack-clusters located behind Berth 1 and Berth 3 drop as the stack-clusters are situated closer to Berth 2. The rgv cycle times of the rgv's subscribed to the stack-clusters behind Berth 2 are more or less constant.

![rgv shuffling cycles](image)

*Figure 7.10, rgv cycle times during shuffling*

The main difference between both graphs is that the rgv cycle times during shuffling are shorter. The reason for these shorter cycle times is that the rgv's working for the stack-cluster behind a portainer location do not depend on one portainer, as is the case when loading a vessel. When shuffling containers the rgv's are spread over the warehouse cranes operating in the stacks, thereby reducing the waiting queues.

The travelling distance between two stacks located in different yards is more or less the same as the distance between a stack and a portainer located on different berths.
7.5 Portainer handling function

The tooth-like portainer handling function can be verified by linking the working times of the portainers to a so-called pointstream, which is a form of output data. During the handling of a feeder vessel all the four portainers have been verified resulting in the graph printed below. This graph shows the following portainer cycle-times during the unloading and loading of a feeder vessel.

![Portainer cycle times graph](image)

*Figure 7.11, the portainer cycle times*

In Figure 7.11 moves 1 to 1250 represent the unloading of the feeder, implying 1250 containers will have to be unloaded, this means each portainer will unload approximately 313 containers.

The unloading starts with the top container from the quayside of the vessel. The time needed to retrieve this container is relatively short, as can be seen in Figure 7.11. After the first container the portainer works horizontally towards the waterside of the vessel, taking away the top layer of containers. After this layer the portainer takes away the second layer. The process of taking away one layer creates one small tooth like shape in the portainer cycle times. In total there are 9 layers in this feeder vessel. The cycle times of the successive layers become higher while the portainer has to go deeper inside the vessel. The three repetitions during the unloading procedure represent the same actions of the portainers after they have moved sideways on the quay to get to another hold.

The second part of Figure 7.11 moves 1267 to 2485 represent the loading part. The small teeth are reversed as a result of the fact that when the portainer starts loading, it starts with the container placed on a location deep in the vessel at the seaside of the feeder-vessel. This figure shows that the shortest cycle time is about 39 seconds and the longest about 77 seconds. The mean cycle time for the loading and unloading of the vessel is 58 seconds.

The portainer cycle times of a mega vessel are similar to the feeder handling times. The only difference is the number of smaller teeth. As a mega vessel is assumed to have 18 horizontal container layers instead of 9.
7.6 Influence on ship cycles when varying the number of rgv’s

In this verification study a calculation is made on the number of needed rgv’s for the loading and the unloading of a vessel. It is expected that when the number of rgv’s per portainer is increased, the dwell times of the vessels at the terminal will become smaller. At a certain amount of rgv’s per portainer the portainer will have a permanent waiting-queue. At this point adding more rgv’s will only result in larger waiting queues saturating the terminal.

A variety of runs have been done in order to check the increase of portainer production, by increasing the number of rgv’s per portainer. Figure 7.12 shows the production of four portainers unloading a feeder vessel moored at Berth 1. During all the runs the mean portainer cycle time was 58 seconds.

![Portainer productivity during the unloading of a feeder vessel](image)

*Figure 7.12 Portainer productivity during the unloading of a feeder vessel*

With one rgv per portainer, the total amount of active rgv’s is four. As the rgv unloading cycle takes about 4 minutes, it is not surprising to see that the production of the four portainers is about 60 moves an hour. Increasing the number of rgv’s per portainer will result in a higher portainer production.

If more than 6 rgv’s per portainer are used to unload the feeder vessel, the portainer production will not increase any further. A surplus of rgv’s will only increase the queue under the portainers.
Figure 7.13 shows the production of four portainers during the adjacent loading of a feeder vessel moored at Berth 1. The containers that are to be loaded are randomly divided over the stacks in Yard 2 situated behind Berth 2.

**Figure 7.13, portainer productivity when adjacent loading using a different number of rgv's**

Figure 7.13 shows the increasing portainer productivity versus the number of rgv's used per portainer. As the rgv cycles during adjacent loading are longer than during unloading, the production of the berth with 1 rgv per portainer is less than during the unloading of the feeder vessel. The berth production with one rgv per portainer is about 25 moves per hour.

When 15 rgv's are used the portainer production is almost 250 containers per four portainers per hour. With an average portainer cycle time of 58 seconds this implies the occupancy rate of the portainers is 100 percent, implying there is always a rgv waiting under the portainer to be served. The consequence of using this much rgv's is the increase of queues under the portainers as can be seen in Figure 7.14 where the x-axis represents the time in seconds.

**Figure 7.14, Portainer 1 queue using 15 rgv's per portainer during the adjacent loading of a feeder**

90
8 Simulation and results

After having verified the reliability of the simulation program in Chapter 7, the results of the simulation runs will be analysed in this chapter. The basis of the simulation runs is the reference terminal as described in Chapter 8.1. The performance indicators will be described in Chapter 8.2. In Chapter 8.3 and 8.4 the reference terminal has been analysed during the handling of one vessel. By handling one vessel at a time the portainers, rgv's and warehouse cranes will not be interfered by rgv's working for other vessels. This way the maximum undisturbed terminal production for the concerning vessel can be determined.

Following these simulation runs the pressure on the terminal has been increased by simulating with three occupied berths in Chapter 8.5. By operating on more than one berth, the production during mutual interference can be determined. Simulation runs concerning two occupied berths have not been carried out. This is based on the assumption that if the achievements of the terminal with 3 occupied berths are sufficient, the achievement of the terminal with 2 occupied berths also is.

8.1 Determination and description of the reference terminal

Before any simulation runs have been made, a standard reference terminal was determined. This means that a standard number of rgv’s, portainers and cranes have been chosen. The performances of these three terminal components have already been determined and illustrated in Chapter 4, but will be included for the sake of completeness. The same applies to the properties of the two types of vessels, as they have been illustrated in Chapter 3.2.

By determining a standard terminal and carrying out the simulation runs, the performances of the terminal can be calibrated, making it possible to analyse variations on the reference terminal in a later stage.

8.1.1 Standardisation of the container vessels

Standardisation of the feeder vessels and the mega vessels simplifies the model making it easier to analyse the performances of the three terminal components and the terminal as a whole.

Characteristics of the vessels

A standard feeder vessel in this simulation has a capacity of 1575 TEU and will be handled by four portainers. Each visit the feeder vessel will be completely unloaded and loaded again, bringing the total amount of handled containers to 3150 TEU. Applying a TEU ratio of 1.5, 2100 moves/call have to be carried out by the portainers during the unloading and the loading of the vessel.

A standard mega vessel has a capacity of 8000 TEU and will be handled by six portainers. The terminal will unload and load 4725 TEU being a 60 % share of the total capacity of the mega vessel. This brings the amount of handled TEU during a mega vessel visit to 9450 TEU. Applying the 1.5 TEU ratio means portainers will have to make 6300 moves per mega vessel visit.
8.1.2 The standardisation of the terminal components

The components on the terminal are the portainers, the horizontal transportation and the container storage system. In the following paragraphs the standardisation of the components on the terminal is dealt with.

*The number of Portainers*
The number of portainers on the quay during the basic runs is determined by the maximum occupation of the berths. When all three the berths are occupied, 14 portainers can be operational. Two feeder vessels moored at Berth 1 and Berth 3 demand four portainers each, while one mega vessel moored at Berth 2 demands 6 portainers.

*Handling performance of the Portainers*
The variable handling performance of the portainers that has been plotted in Figure 7.11 in Chapter 7.5 is the standard handling performance of the portainers.

*The attributes of the Rgv’s*
The properties of the rgv, acceleration, speed and wheel turning have been determined in Chapter 4. Standard value for acceleration is 1 m/s². Standard high speed on the berth lanes and the yard lanes is 3 m/s and standard low speed on the warehouse lanes is 1 m/s. The time needed for a rotation disk to turn the wheels of an rgv is 4 seconds.

*The number of Rgv’s per portainer*
The standard number rgv’s subscribed to a portainer during unloading and fast loading differs from the number of rgv’s subscribed to a portainer during adjacent loading and shuffling. In the first case 6 rgv’s per portainer is the standard amount, while in the latter case 10 rgv’s per portainer is standard. These numbers are based on the rgv cycle times as determined in Appendix 10 and the necessity to keep the portainers from waiting for rgv’s.

*Number of Warehouse cranes*
The standard number of warehouse cranes equals the maximum number of warehouse cranes possible on the terminal. This amounts to 70 warehouse cranes of which 30 warehouse cranes are located in the stacks behind Berth 2. Berth 1 and Berth 3 each host 20 warehouse cranes. After the simulation runs the occupancy rates of the warehouse cranes will be analysed.

*Handling performance of the Warehouse cranes*
As opposed to the portainers, the warehouse cranes have an invariable handling performance in the simulation program. The warehouse crane performance is determined by the average time that a crane needs to work in order to retrieve a container from the stack and delivering it back to a service point. This average crane cycle is calculated in Chapter 4.3 and is 80 seconds. One simulation run will be made by using variable crane cycles.
8.2 Performance indicators

The simulation runs with one and three vessels will be evaluated using a number of performance indicators. A wide variety of simulation runs will be carried out during each of the two types of berth occupation. The simulation of one feeder vessel moored at Berth 1, for instance, consists of four separate runs. The runs simulate unloading, fast loading, adjacent loading and shuffling. By performing a separate run for each mode the output of each simulation will be easier to analyse.

The performance indicators that will be used for this purpose are listed below:

- Average container production per berth
- Portainer occupation rates
- Warehouse crane occupation rates
- Rgv rows on crossing nodes
- Rgv rows on the warehouse lanes
- Rgv rows under the portainers

Average container production per berth
By determining the container production during each run a first impression can be obtained on the performance of the terminal during each mode. The container production during a run with a high berth occupation can be compared to runs with a single vessel.

Portainer occupation rates
Apart from shuffling, all the vessel handling modes make use of portainers. The portainer occupancy therefore is an important performance indicator. The portainer occupancy rate must be kept as high as possible. When the average portainer cyletimes stay the same, the portainer occupancy rate is directly linked to the portainer production.

Warehouse crane occupancy rates
The warehouse crane occupancy rate is especially important for the determination of the number of cranes in the stack.

Rgv rows on crossing nodes
Rgv rows should be minimised.

Rgv rows on the warehouse lanes
The length of the rgv row on a warehouse lane in a warehouse may not exceed the number of 3 rgv's because there is no more room available in a stack-row.

Rgv rows under the portainers
Rgv rows under portainer may preferably not exceed 6 rgv's, as there is not enough room for more. This number can only be exceeded if additional lanes are added to the layout.
8.3 Feeder vessel moored at Berth 1

With just one vessel present at the terminal and no interference from rgv activities operating for other berths the handling productivities will turn out to be relative high. First a feeder vessel moored at Berth 1 will be analysed during the different handling procedures, followed by the handling procedures of a mega vessel moored at Berth 2.

![Feeder at berth 1](image)

*Figure 8.1, a feeder vessel moored at Berth 1*

The next simulation runs have been carried out with one feeder vessel moored at Berth 1. Berth 2 and Berth 3 remain empty. The undisturbed simulation runs for a feeder vessel, moored at Berth 1 are assumed to be similar to those of a feeder vessel moored at Berth 3.

The four experiments with a feeder vessel moored at Berth 1 are:

- Unloading of a feeder
- Fast-loading of a feeder
- Loading of a feeder from an adjacent berth
- Shuffling from a feeder berth

The basic parameters used for each of the experiments are shown in Appendix 13-15.
8.3.1 Unloading of a feeder vessel

This simulation run concerns the unloading of 1050 containers from the feeder vessel to the stack-clusters located behind Berth 1. In Appendix 13 the rgv waiting rows under the portainers, at the nodes and on the warehouse lanes can be seen.

![Diagram of yard layout](image)

*Figure 8.2, the unloading of a feeder vessel*

*Portainer occupancy rates*

The portainer occupancy rates during the unloading of the feeder vessel are shown in Figure 8.3. These occupancy rates will act as a comparison with other experiments. Each portainer has 6 rgv's subscribed during the unloading of the feeder vessel.

![Occupancy rates graph](image)

*Figure 8.3, The portainer occupancy rates during the unloading of a feeder vessel at Berth 1*

The occupancy rate of the odd numbered portainers is higher than the occupancy rate of the even numbered portainers. This is a result of the relative longer distance that the rgv's have to cover at the even portainer locations resulting in a longer rgv driving cycle.

*Handling production*

The handling production of the portainers located at Berth 1 during the unloading of one feeder vessel is 237 moves per hour. This results in 355 TEU per hour. The handling time for a feeder vessel during unloading is 4.4 hours.
Rgv waiting queues under the portainers
Figure 8.4 shows the waiting queues of the rgv's under each portainer. The portainer occupancy rates plotted in Figure 8.3 are very high, which agrees with the fact that the rgv queue under the portainers is seldom empty.

![Rgv waiting queues under the portainers during the fast loading of a feeder vessel moored at berth 1](image)

*Figure 8.4, rgv queue under the portainers at Berth 1 during the unloading of a feeder*

The rgv waiting queues under the portainers almost never exceed the length of 4 rgv's. The average queue length is 1.5 rgv's.

Rgv waiting queues on the warehouse lanes
The figures of the queues on the warehouse lanes show no extraordinary results. The queues almost never exceed 1 rgv in length. The queues on the warehouse lanes are given in Appendix 13. An example of a rgv waiting queue on a warehouse lane during the unloading and subsequently the fast loading of the feeder vessel can be seen in Figure 8.5.

![Rgv waiting queues on warehouse lane 22](image)

*Figure 8.5, rgv waiting queues on warehouse lane 22*

Warehouse crane occupancy rates
The warehouse crane occupancy rates for the warehouse cranes located in Yard 1 have been calculated in Appendix 13, resulting in:

Mean occupancy rate for a warehouse crane located in the 20' stack is 33.1%.
Mean occupancy rate for a warehouse crane located in the 40' stack is 22.1 %.

These occupancy rates are acceptable. It might be possible to operate with fewer warehouse cranes.
8.3.2 Fast loading of a feeder vessel

This simulation run concerns the fast loading of 1050 containers onto the feeder vessel from the stack-clusters located behind Berth 1. The fast loading of a feeder vessel is only possible if the containers have been shuffled from Yard 2 to Yard 1 earlier. In Appendix 13 the rgv waiting rows concerning the portainers, nodes and warehouse lanes can be seen.

![Diagram of port loading](image)

*Figure 8.6, the fast loading of a feeder vessel*

**Portainer occupancy rates**

If a feeder vessel is being loaded with containers from the yard behind the berth the feeder vessel is moored to, the rgv's use the same routes as during the unloading of a feeder vessel. The expected portainer productions will therefore be more or less the same. The portainer occupancy rates are given in Figure 8.7.

![Portainer occupancy rates chart](image)

*Figure 8.7, the portainer occupancy rates during the fast loading of a feeder vessel*

**Handling production**

The handling production of the portainers located at Berth 1 during the fast loading of one feeder vessel is 244 moves per hour. This results in 366 TEU per hour. The handling time for a feeder vessel during unloading is 4.3 hours. The portainer productions are a bit higher than when unloading a feeder vessel.
Rgv waiting queues under the portainers

Figure 8.8 shows the waiting queues of the rgv’s under each portainer. The portainer occupancy rates plotted in Figure 8.7 are high, which agrees with the fact that the rgv queue under the portainers is seldom empty.

![Graph showing percentage of handling time vs. row length for different portainers](image)

*Figure 8.8, rgv waiting rows under portainers at Berth 1 when loading fast*

Warehouse crane occupancy rates

The warehouse crane occupancy rates for the warehouse cranes located in Yard 1 have been calculated in Appendix 13, resulting in:

Mean occupancy rate for a warehouse crane located in the 20’ stack is 33.9 %.
Mean occupancy rate for a warehouse crane located in the 40’ stack is 22.6 %.

These occupancy rates seem acceptable. It might even be possible to operate with fewer warehouse cranes.
8.3.3 Adjacent loading of a feeder

The adjacent loading of a feeder vessel moored at Berth 1 means that the 1050 to be loaded containers have to be retrieved from Yard 2. 10 rgv's per portainer will be used for this. In this run the node rows on the terminal have been analysed, as the rgv's are to travel between the berths. In Appendix 14 the rgv waiting rows concerning the portainers, nodes and warehouse lanes can be seen for this run.

![Diagram of yard and berths](image)

Figure 8.9, the adjacent loading of a Feeder

Portainer occupancy rates

The portainer occupancy rate during the adjacent loading of a feeder vessel is higher as the portainers are situated closer to Yard 2. This is a result of the shorter driving distances of the rgv’s.

![Bar chart of portainer occupancy rates](image)

Figure 8.10, the portainer occupancy rates when loading a feeder from an adjacent berth

Higher portainer occupancy rates might be achieved by increasing the number of rgv’s per portainer. Especially Portainer 1 & 2 will have their production increased by this measure.

Handling production

The portainer production is 51.5 moves per portainer per hour during adjacent loading. This results in a berth production of 206 moves per hour. This results in 309 TEU per hour per feeder berth. The handling time needed for the adjacent loading of 1050 containers is 5.1 hours.
Rgv waiting queues under the portainers
The length of the waiting queues under the portainers rarely exceeds the length of 4 rgv's, as can be seen in Figure 8.11.

Rgv waiting queues under portainer at Berth 1 during adjacent loading

![Chart showing percentage of handling time against row length for different portainers]

*Figure 8.11, the rgv waiting queues under the portainers at Berth 1 when adjacent loading*

At the portainer locations 1 and 2, the waiting rows will be empty for about 20 percent of the time. In this case it could be an option to work with more rgv's per portainer.

Rgv waiting queues at the nodes
The nodes containing the longest rgv queues are the nodes 116, 114, 260, 290, 243 and 273. The queues at these nodes are shown in Appendix 14. The length of the queues at these nodes is generally 1 or 2 rgv's. A node queue rarely contains 3 rgv's.

Rgv waiting queues on the warehouse lanes
The queues on the warehouse lanes in yard 2 are at the most 2 rgv's in length but most times just 1. These queues are shown in appendix 14.

Warehouse crane occupancy rates
The warehouse crane occupancy rates for the warehouse cranes located in Yard 2 have been calculated in Appendix 14 resulting in:

Mean occupancy rate for a warehouse crane located in the 20' stack is 19.1 %.
Mean occupancy rate for a warehouse crane located in the 40' stack is 12.7 %.

These occupancy rates seem acceptable. It might even be possible to operate with fewer warehouse cranes.
8.3.4 Shuffling from a feeder berth

When the feeder vessel has been unloaded and loaded again the rgv's operating for the portainers on Berth 1 will be subscribed to the stack-clusters behind their portainers. The containers delivered by the feeder vessel can then be shuffled from Yard 1 to Yard 2. The number of shuffled containers is 1250.

When the rgv unloads a container from Yard 1 into Yard 2 the rgv will wait to be loaded with a container that can be shuffled from Yard 2 to Yard 1, if there is any. Otherwise it will return empty to retrieve the next shuffle container from Yard 1. In Appendix 15 the rgv waiting rows concerning the nodes and warehouse lanes can be seen for this run.

![Diagram showing shuffling from Yard 1 to Yard 2 and Yard 3]

Figure 8.12, the shuffling from Yard 1

Shuffle production
During this simulation run 1250 containers have been placed equally divided over the 20' stack and the 40' stack in Yard 1. The same amount of containers has been placed in Yard 2.

The shuffle production is defined as the total amount of shuffled containers per time unit. The total amount of shuffled containers is:

- Number of shuffled containers from Yard 1 to Yard 2: 1,250
- Number of shuffled containers from Yard 2 to Yard 1: 1,250
- Total amount of shuffled containers on the terminal: 2,500

The time necessary to shuffle this amount of containers is 26,670 seconds or 7.4 hours.

This results in a total shuffle production of: \[ \frac{2,490}{7.4} = 337.8 \] containers per hour, or 5.6 containers per minute.

Rgv waiting queues at the nodes
The nodes with the highest expected rgv rows have been analysed in order to get an impression on the busyness on the terminal during the shuffling procedure. The nodes are the same ones as during the adjacent loading of a feeder vessel. The nodes are nodes 116, 114, 260, 290, 243 and 273. The waiting rows of these nodes have been plotted in Appendix 15. What can be seen is that length of the rgv queues occasionally exceeds 2 rgv's. The node with the longest queue has been plotted in Figure 8.13.
Waiting rows on the warehouse lanes
The warehouse lanes in the warehouses located in Yard 1 and Yard 2 have been analysed. The warehouse lanes in Yard 1 however will be visited more frequently as the number of warehouses is smaller compared to Yard 2.

Warehouse lane 21 is located in Yard 1. Despite the frequent visits of the rgv’s the row length does not reach 3 rgv’s.

Warehouse crane occupancy rates
The warehouse crane occupancy rates for the warehouse cranes located in Yard 1 are the highest on the terminal during the shuffling from Berth 1. The occupancy rates of these warehouse cranes have been calculated in Appendix 15, resulting in:

Mean occupancy rate for a warehouse crane located in the 20’ stack in Yard 2 is 46.7 %
Mean occupancy rate for a warehouse crane located in the 40’ stack in Yard 2 is 31.2 %

These occupancy rates are acceptable.
8.3.5 Recapitulation of the feeder vessel simulation runs

Handling rates
The performance of the terminal during the handling of a feeder vessel moored at Berth 1 without any disturbances from activities on other berths has been plotted in Table 8.1.

<table>
<thead>
<tr>
<th></th>
<th>Number of containers</th>
<th>Production per portainer [moves/hour]</th>
<th>Production per berth [moves/hour]</th>
<th>Production per berth [TEU/hour]</th>
<th>Service times [hours]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unloading</td>
<td>1050</td>
<td>59</td>
<td>236</td>
<td>355</td>
<td>4.5</td>
</tr>
<tr>
<td>Fast loading</td>
<td>1050</td>
<td>61</td>
<td>244</td>
<td>366</td>
<td>4.3</td>
</tr>
<tr>
<td>Adjacent loading</td>
<td>1050</td>
<td>51</td>
<td>206</td>
<td>309</td>
<td>5.1</td>
</tr>
</tbody>
</table>

*Table 8.1, the undisturbed container productions for Berth 1*

The assumption that the unloading of a feeder vessel can be carried out with comparable speed as during the fast loading procedure has proven to be correct. The adjacent loading of a feeder vessel takes about 13 % longer than fast-loading with this terminal configuration. The most time consuming unload-load combination for the feeder vessel is the unloading combined with the adjacent loading strategy. The undisturbed service time for the handling of a feeder vessel is: \( 4.5 + 5.1 = 9.6 \) hours.

Shuffling containers from Yard 2 to Yard 1 allows the feeder vessel to be fast-loaded. The service time will then be \( 4.5 + 4.3 = 8.8 \) hours. This is a reduction of 0.8 hours or 8 % of the total service time compared to the adjacent loading variant. The production during shuffling has been determined to amount to 337.8 containers per hour. This is based on shuffling 2 containers per rgv cycle. The first container is shuffled from Yard 1 to Yard 2 while the second will be shuffled from Yard 2 to Yard 1. This makes the determined shuffling production an optimistic one. Even so, it is questionable if a reduction of 0.8 hours in the service times will justify the shuffling procedure.

Operational rgv's - Portainer occupancies
The amount of operational rgv's on the reference terminal during the different procedures has been established in Chapter 5 to be 6 rgv's per portainer for the unloading and fast-loading procedure and 10 rgv's per portainer for the adjacent loading procedure. Having carried out the simulation runs it can be concluded that the chosen 6 rgv's for the unloading and fast-loading procedures results in high portainer occupancy rates of around 95 % without creating excessive waiting rgv rows in the stacks, nodes or under the portainers. Row length under the portainers can be seen in Appendix 13-15. These row lengths seldom exceed the length of 5 rgv's during any of the simulated handling procedures.

The adjacent loading procedure with 10 rgv's per portainer, results in a mean portainer occupancy rate of 83 % on Berth 1. Increasing the number of rgv's per portainer to 12 should increase the occupancy rates and reduce the service time of a feeder vessel, thereby further justifying the criticism on the shuffling procedure.
**Warehouse crane occupancy rates**
The occupation rates of the warehouse cranes during the undisturbed handling of a feeder vessel have been shown in Table 8.2.

<table>
<thead>
<tr>
<th>Stack type</th>
<th>Unloading to Yard 1 (%)</th>
<th>Fast-loading from Yard 1 (%)</th>
<th>Adjacent loading from Yard 2 (%)</th>
<th>Shuffling from Yard 1 (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>20' stack</td>
<td>33.1</td>
<td>33.9</td>
<td>19.1 (Y2)</td>
<td>46.7 (Y1)</td>
</tr>
<tr>
<td>40' stack</td>
<td>22.1</td>
<td>22.6</td>
<td>12.7 (Y2)</td>
<td>31.3 (Y1)</td>
</tr>
</tbody>
</table>

*Table 8.2, warehouse crane occupation rates during the undisturbed handling of a feeder vessel*

The warehouse crane occupancy rates are low. During the undisturbed handling of a feeder vessel it is recommend operating with less warehouse cranes than have been introduced on the reference terminal. In Chapter 8.5 more definite conclusions can be made as that chapter concerns the handling of 3 vessels moored at the terminal.

**Rgv waiting queues on the nodes**
Rgv row length on the nodes during adjacent loading has been plotted in Appendix 14. What can be concluded is that during that simulation run the row lengths on the nodes never exceeds the length of 2 rgv's.
8.4 Experiment: Mega vessel moored at Berth 2

![Diagram of Mega vessel at berth 2]

Figure 8.15, mega vessel moored at Berth 2.

When there is no disturbance from other activities on the terminal the handling productions are expected to be high. The next simulation runs have been carried out with one mega vessel moored at Berth 2. Berth 1 and Berth 3 remain empty.

The four experiments with the mega vessel are:
- Unloading of a mega vessel
- Fast-loading of a mega vessel
- Loading of the mega vessel from the adjacent berths
- Shuffling from Berth 2

The basic parameters used during these runs are as shown in Appendix 16-19.
8.4.1 The unloading of the mega vessel

This simulation run concerns the unloading of a mega vessel moored at Berth 2. 3150 containers will be unloaded and stacked in Yard 2, located behind Berth 2. In Appendix 16 the rgv waiting rows concerning the portainers, nodes and warehouse lanes can be seen.

![Diagram of yard layout](image)

*Figure 8.16, the unloading of a mega vessel*

**Portainer occupancy rates**

The portainer occupancy rates during the unloading of a mega vessel using 6 rgv's per portainer are shown in Figure 8.17. These occupancy rates will act as a comparison for other experiments using a mega vessel in the future.

![Portainer occupancy rates chart](image)

*Figure 8.17, the occupancy rate of portainers 5 to 10*

The occupancy rate of the odd numbered portainers is higher than the occupancy rates of the even numbered portainers. This is a result of the relative longer distance the rgv's have to cover when subscribed to the even numbered portainers resulting in a longer rgv driving cycle.

**Handling production**

The handling production of the portainers located at Berth 2 during the unloading is 58.7 moves per portainer per hour. This results in 352 moves per berth per hour or 528 TEU per berth per hour. The handling time for the unloading of a mega vessel is 8.9 hours.
Rgv waiting queues under the portainers
Figure 8.18 shows the waiting queues of the rgv's under each portainer. The portainer occupancy rates plotted in are very high, which agrees with the fact that the rgv queue under the portainer is seldom empty.

![Rgv waiting queues under the portainers at berth 2 during the unloading of a mega vessel](image)

*Figure 8.18, rgv waiting queues under the portainers on Berth 2*

The rgv waiting queues under the portainers almost never exceed the length of 4 rgv's. The average queue length is 1.55 rgv's.

Rgv waiting rows on the warehouse lanes
The figures of the queues on the warehouse lanes show no extraordinary results. The queues almost never exceed 1 rgv in length. The queues on the warehouse lanes are given in Appendix 16.

Warehouse crane occupancy rates
The warehouse crane occupancy rates for the warehouse cranes located in Yard 2 have been calculated in Appendix 16, resulting in:

Mean occupancy rate for a warehouse crane located in a 20' stack is 32.8%
Mean occupancy rate for a warehouse crane located in a 40' stack is 21.8%

These occupancy rates seem acceptable. It might even be possible to operate with fewer warehouse cranes. The queues on some of the warehouse lanes are given in Appendix 16.
8.4.2 The fast loading of a mega vessel

This simulation run concerns the fast loading of 3150 containers onto a mega vessel from Yard 2. The routes of the rgv's are almost the same as during unloading. The results are expected to be comparable. In Appendix 17 the rgv waiting rows concerning the portainers, nodes and warehouse lanes can be seen.

![Diagram of Yards and Berths](image)

*Figure 8.19, the fast loading of a mega vessel*

**Portainer occupancy rates**

In Figure 8.20 the portainer occupancy rates when fast loading a mega vessel are given. The difference in driving distance of the rgv’s subscribed to the even numbered and odd numbered portainers is apparent.

![Bar chart of Portainer occupancy rates](image)

*Figure 8.20, Portainer occupancy rates during the fast loading of a mega vessel.*

**Handling production**

The production of Berth 2 during the fast loading of 3150 containers onto a mega vessel is 350 moves per hour. This results in 525 TEU per berth per hour. The required handling time is 9 hours.

**Rgv waiting queues under the portainers**

Figure 8.21 shows the waiting queues under the portainers during the fast loading the mega vessel. The waiting queues under the portainer are seldom empty.
Figure 8.21, the rgv waiting queues under the portainers when loading a mega vessel

Warehouse crane occupancy rates
The warehouse crane occupancy rates for the warehouse cranes located in Yard 2 have been calculated in Appendix 17, resulting in:

Mean occupancy rate for a warehouse crane located in a 20' stack is 32.8%
Mean occupancy rate for a warehouse crane located in a 40' stack is 21.8%

These occupancy rates seem acceptable. The queues on some of the warehouse lanes are given in Appendix 17.
8.4.3 Adjacent loading of a mega vessel

The adjacent loading of a mega vessel moored at Berth 2 means 3150 containers will have to be retrieved from Yard 1 & 3. The rgv waiting rows on the nodes on the terminal have been analysed, as the rgv's travel between the yards. In Appendix 18 the rgv waiting rows concerning the portainers, nodes and warehouse lanes can be seen for this run.

![Diagram of adjacent loading of a mega vessel](image)

*Figure 8.22, the adjacent loading of a mega vessel*

**Portainer occupancy rates**
During this simulation run 10 rgv's per portainer are used.

![Bar chart of portainer occupancy rates](image)

*Figure 8.23, Portainer occupancy rate when loading a mega vessel from adjacent berths*

**Handling production**
The production per portainer is 51 moves. Per berth this results in 305 moves per hour. This results in 458 TEU per hour per berth. The handling time needed for the undisturbed adjacent loading is 10 hours.
Rgv waiting queues under the portainers

Figure 8.24 shows that the rgv waiting queues under the portainers sometimes exceed the length of 4 rgv's.

![Rgv waiting queue under the portainer during the adjacent loading of a mega vessel](image)

Figure 8.24, the rgv waiting queues under the portainers during the loading from an adjacent berth

At all the portainer locations the waiting row is empty about 13% of the time. In this case it could be an option to work with more rgv's per portainer.

Rgv waiting queues at the nodes

The nodes containing the longest rgv queues are the nodes that connect the inter-berth lanes with the stacks. The queues at these nodes are shown in Appendix 18.

Warehouse crane occupancy rates

The warehouse crane occupancy rates for the warehouse cranes located in Yard 2 have been calculated in Appendix 18, resulting in:

- Mean occupancy rate for a warehouse crane located in a 20' stack is 22.6%
- Mean occupancy rate for a warehouse crane located in a 40' stack is 15.0%

These occupancy rates seem acceptable. The queues on some of the warehouse lanes are given in Appendix 18.
8.4.4 The shuffling from Berth 2

When the mega vessel has been unloaded and loaded again the rgv’s operating for the portainers on Berth 2 will be subscribed to the stack-clusters behind their portainers. The 3150 containers delivered by the mega vessel will then be shuffled from Yard 2 to Yard 1 & 3. When the rgv transports a container from Yard 2 into Yard 1 or Yard 3, the rgv will wait to be loaded by a container that can be shuffled from Yard 1 or 3 to Yard 2, if there is any. Otherwise it will return empty to retrieve the next shuffle container from Yard 2. In Appendix 19 the rgv waiting rows concerning the nodes and warehouse lanes can be seen for this run.

![Diagram of shipyard layout](image)

*Figure 8.25, shuffling from Berth 2*

**Shuffle productions**

The shuffle production is defined as the total amount of shuffled containers per time unit. The shuffle procedure lasted 720 minutes, 11.25 hours. In this time interval rgv’s moved 1575 containers from Yard 2 to Yard 1, 1575 containers from Yard 2 to Yard 3 and 3150 containers from Yard 1 and 3 to Yard 2. In total 6300 containers were put in their place.

This results in a total shuffle production of: $6300/11.25 = 560$ containers per hour, or 9.3 containers per minute.

**Rgv waiting queues at the nodes**

If a mega vessel has left Berth 2, the shuffling procedure can start. While shuffling from Berth 2 to the adjacent berths 10 rgv’s per stack-cluster are used. In this experiment $10 \times 6 = 60$ rgv’s are used. It is expected that this should be no problem for the terminal and no excessive queues should develop at any node or stack. The queues at the nodes almost never exceed the length of 3 rgv’s except at node 363 and 393. These nodes are shown in Figure 8.26 and Figure 8.27. These nodes are the connections between the bottom inter berth lanes and the yard lanes of Yard 2. As can be seen in Appendix 2.

![Graph of rgv queue](image)

*Figure 8.26, the rgv queue at node 363 when shuffling from Berth 2*
Some other nodes are shown in Appendix 19.

The queues on the warehouse lanes
In this experiment in total 60 shuffle rgv's get their containers from 80 warehouse lanes, so no problems are expected on the warehouse lanes. The queues on the warehouse lanes almost never exceed the total of 2 rgv's. Warehouse lane 23 is an exception the queues on this warehouse lane can be seen in Figure 8.28. The queues of stack 21 to 30 are shown in Appendix 19.

Warehouse crane occupancy rates
The warehouse crane occupancy rates for the warehouse cranes located in Yard 2 are the highest on the terminal during the shuffling from Berth 2. The occupancy rates of these warehouse cranes have been calculated in Appendix 19, resulting in:

Mean occupancy rate for a warehouse crane located in the 20' stack in Yard 2 is 51.9 %
Mean occupancy rate for a warehouse crane located in the 20' stack in Yard 2 is 34.6 %

These occupancy rates are acceptable.
8.4.5 Recapitulation of the mega vessel simulation runs

Handling rates
The performance of the terminal during the handling of a mega vessel moored at Berth 2 without any disturbances from activities on other berths has been plotted in Table 8.3.

<table>
<thead>
<tr>
<th></th>
<th>Number of containers</th>
<th>Production per berth [moves/hour]</th>
<th>Production per portainer [moves/hour]</th>
<th>Production per berth [TEU/hour]</th>
<th>Service times [hours]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unloading</td>
<td>3150</td>
<td>352</td>
<td>58</td>
<td>528</td>
<td>8.9</td>
</tr>
<tr>
<td>Loading</td>
<td>3150</td>
<td>350</td>
<td>58</td>
<td>525</td>
<td>9</td>
</tr>
<tr>
<td>Adjacent loading</td>
<td>3150</td>
<td>324</td>
<td>54</td>
<td>486</td>
<td>9.7</td>
</tr>
</tbody>
</table>

*Table 8.3, berth productions for Berth 2*

As was the case with the feeder vessel, the assumption that the unloading of a feeder vessel can be carried out with comparable speed as the fast loading has proven to be correct. The adjacent loading of a mega vessel takes about 8% longer than during fast loading with this terminal configuration. The most time consuming unload-load combination for the mega vessel is the unloading combined with the adjacent loading strategy. The undisturbed service time for the handling of a mega vessel is \(8.9 + 0.7 = 10.6\) hours.

Shuffling containers from Yard 1 & 3 to Yard 2 will allow the mega vessel to be fast-loaded. The service time will then be \(8.9 + 9 = 17.9\) hours. This is a reduction of 0.7 hours or 4% compared to the handling variant with adjacent loading. The production during shuffling has been determined to amount to 560 containers per hour. This is based on shuffling 2 containers per rgv cycle. The first container is shuffled from Yard 2 to Yard 1 or 3 while the second will be shuffled back to Yard 2 from Yard 1 or Yard 3. This makes the determined shuffle production an optimistic one. Even so, it is questionable if a reduction of 0.7 hours in the service times will justify the shuffling procedure.

Operational rgv's - Portainer occupancies
The amount of operational rgv's on the reference terminal during the different handling procedures for a mega vessel has been established to be 6 rgv's per portainer for the unloading and fast-loading procedure and 10 rgv's per portainer for the adjacent loading procedure in Chapter 5. Having carried out the simulation runs it can be concluded that the chosen 6 rgv's for the unloading and fast-loading procedures results in high portainer occupancy rates without creating excessive rgv waiting rows in the stacks or on the nodes. The adjacent loading procedure with 10 rgv's per portainer however results in a mean portainer occupancy rate of about 83% on Berth 2. Increasing the number of rgv's per portainer to 12 should increase the occupancy rates and reduce the service time of a mega vessel. Obviously bigger queues under the portainers can be expected.
**Warehouse crane occupancy rates**
The occupation rates of the warehouse cranes during the undisturbed handling of a mega vessel have been shown in Table 8.4.

<table>
<thead>
<tr>
<th>Stack type</th>
<th>Unloading to Yard 2 (%)</th>
<th>Fast-loading from Yard 2 (%)</th>
<th>Adjacent loading from Yard 1 or 3 (%)</th>
<th>Shuffling from Yard 2 (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>20’ stack</td>
<td>32.8</td>
<td>32.8</td>
<td>22.6 (Y1 or Y3)</td>
<td>51.9 (Y2)</td>
</tr>
<tr>
<td>40’ stack</td>
<td>21.8</td>
<td>21.8</td>
<td>15.0 (Y1 or Y3)</td>
<td>34.6 (Y2)</td>
</tr>
</tbody>
</table>

*Table 8.4, warehouse crane occupation rates during the undisturbed handling of a mega vessel*

The warehouse crane occupancy rates are low. During the undisturbed handling of a mega vessel it is recommend operating with less warehouse cranes than have been introduced on the reference terminal. In Chapter 8.5 more definite conclusions can be made as that chapter concerns the handling of 3 vessels moored at the terminal.

**Rgv waiting queues on the nodes**
The rgv row length on the nodes during adjacent loading has been plotted in Appendix 18. What can be concluded is that during that simulation run the row lengths on the nodes occasionally amounts to the length of 3 rgv’s, but never 4 rgv’s.
8.5 Simulation runs with 3 occupied berths

In the previous chapter the terminal has been tested during undisturbed conditions, with one moored vessel at the terminal. During this configuration no problems have been encountered. In this chapter the pressure on the terminal has been increased by having 3 vessels moored at the berths. Numerous combinations of loading and unloading strategies can be worked out, but only the 4 that cause the highest pressure on the terminal have been simulated.

![Diagram of berths]

*Figure 8.29, all the berths occupied*

The next experiments have been carried out with all the berths occupied. The basic parameters used during these runs are shown in Appendix 20-23. The experiments are listed in the Table 8.5.

<table>
<thead>
<tr>
<th>Experiment</th>
<th>Berth 1</th>
<th>Berth 2</th>
<th>Berth 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Adjacent loading</td>
<td>Adjacent loading</td>
<td>Adjacent loading</td>
</tr>
<tr>
<td>2.</td>
<td>Adjacent loading</td>
<td>Unloading</td>
<td>Adjacent loading</td>
</tr>
<tr>
<td>3.</td>
<td>Unloading</td>
<td>Adjacent loading</td>
<td>Unloading</td>
</tr>
<tr>
<td>4.</td>
<td>Shuffling</td>
<td>Shuffling</td>
<td>Shuffling</td>
</tr>
</tbody>
</table>

*Table 8.5, Experiments with 3 occupied berths*
8.5.1 Experiment 1

This simulation run concerns the simultaneous adjacent loading of the three moored vessels. This situation may occur when all the berths are occupied and the pre-shuffling of containers has not taken place on the terminal. In this situation the nodes located on, or giving access to the inter-berth lanes will be tested at their maximum. The space that has to be reserved for waiting rows on the inter-berth lanes has been determined during this simulation run. For this reason also nodes located on the outer inter-berth lanes have been included in the output.

![Diagram showing adjacent loading on all berths](image)

*Figure 8.30, adjacent loading on all berths*

**Portainer occupancy rates**

As can be seen in Figure 8.31, the occupancy rate of the portainers located on the feeder vessel berths, Portainer 1-4 and Portainer 11-12 are slightly lower than the portainer occupancy rates of the portainers located on Berth 2. The reason is that the rgv driving distances decrease from the Portainer 1 to Portainer 4 as the rgv's working for both portainers retrieve their containers from Yard 2. This also happens on Berth 3. As the rgv's working for the portainers located on Berth 2 retrieve their containers from both Yard 1 and Yard 2, the rgv driving distances are more constant, which agrees with the figure and results in higher portainer occupancy rates.

**Handling production**

The above described portainer occupancy rates result in productions per berth as shown in Table 8.4. These results agree with what might be suspected from the portainer occupancy rates. The mean portainer productions of Berth 1 and Berth 3 are lower than the mean portainer capacity of 60 moves per hour.
<table>
<thead>
<tr>
<th></th>
<th>Number of adjacent loaded containers</th>
<th>Production per portainer (moves/hour)</th>
<th>Production per portainer (moves/hour)</th>
<th>Production per berth (TEU/hour)</th>
<th>Service time (hours)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Berth 1</td>
<td>1050</td>
<td>48.6</td>
<td>194.3</td>
<td>291.6</td>
<td>5.4</td>
</tr>
<tr>
<td>Berth 2</td>
<td>3150</td>
<td>54.8</td>
<td>329.1</td>
<td>493.7</td>
<td>9.7</td>
</tr>
<tr>
<td>Berth 3</td>
<td>1050</td>
<td>48.6</td>
<td>195.5</td>
<td>293.3</td>
<td>5.4</td>
</tr>
</tbody>
</table>

Table 8.6, berth productions during adjacent loading on all berths

The handling times of the feeder vessels during adjacent loading on the terminal with three occupied berths is only 0.3 hour higher than the handling times during adjacent loading of only one moored feeder vessel.

This result can be defended by realising that the rgv’s working for Berth 2 take the first inter-berth lane ring and that the rgv’s working for Berth 1 & 3 take the second inter-berth lane ring thereby separating the inter-berth rgv flow. The fact that rgv’s belonging to separate berths also retrieve their containers from separate yards means that the only place where the rgv flows will interact is at the entrances of the yard-lanes and on crossing points on the inter-berth lanes.

The fact that the rgv’s working for Berth 1 & Berth 3 have to take the second ring means that they will always have to give the right of way to the rgv’s working for Berth 2 that take the first ring. This is the reason that the service times on Berth 2 do not differ much from the undisturbed situation and that the service times on Berth 1 & 3 take 0.3 hours longer than the undisturbed situation.

The handling times of the mega vessels during adjacent loading on the terminal with three occupied berths is the more or less the same as the handling times during the undisturbed adjacent loading of only one moored mega vessel.

**Rgv waiting queues under the portainers**

The occupancy rates depend on the availability of rgv’s waiting in the portainer queue. If the queue is empty the portainer will not be able to dispose of its container which is unfavourable for the portainer occupancy rate. A steady flow of rgv’s is needed to keep the portainer occupancy rates high.

![Figure 8.32, rgv waiting rows under the portainers on Berth 1](image)

Under the portainers on Berth 1 the rgv waiting rows have not exceeded a number of 5 rgv’s, as can be seen in Figure 8.32. Most of the time the waiting row under the portainers is 1 rgv. For about 20 % of the time no rgv’s were present resulting in inactivity periods for the portainers. A solution could be to increase the number of rgv’s operating for the portainers on Berth 1.
During the adjacent loading of the mega vessel moored at Berth 2, the rgv row has amounted to 6 rgv's only once. For the longest part of the time the rgv-waiting row has been 1 rgv. The rgv rows under the portainers where empty for about 12.5% of the time, which is lower than the 20% of Berth 1 and Berth 3. This is consistent with the fact that the production on Berth 2 is higher due to the shorter inactivity periods of the portainers. The rgv waiting rows under the portainers on Berth 2 can be seen in Figure 8.33.

![Rgv waiting queues under the portainers on Berth 2](image)

The rgv waiting rows under the portainers located on Berth 3 are comparable to the waiting rows on Berth 1. This can be seen in Figure 8.34.

![Rgv waiting queues under the portainers on Berth 3](image)

**Rgv waiting queues at the nodes**
The nodes with the expected highest rgv rows have been analysed in order to get an impression on the busyess of the terminal during the adjacent loading of three vessels. In Appendix 7 the rgv flows on the terminal have been calculated for this situation. From that calculation the nodes most liable to suffer long rgv rows have been identified. These are the nodes 110, 350, 108, 380 and 106 that connect the top inter-berth lanes with the yard-lanes. The nodes 508, 333, 510, 363 and 512 connect the bottom inter-berth lanes with the yard-lanes. A number of nodes located on more outward situated inter-berth lanes have been included in order to determine the needed space for rgv waiting rows. The waiting rows of these nodes have been plotted in Appendix 20. What can be seen is that only the length of the rgv queue on node 350 reaches 6 rgv's once. The maximum rgv length on the nodes does not exceed the length of 4 except on node 350.
Rgv waiting rows on the warehouse lanes
The warehouse lanes of the warehouses in Yard 2 are the busiest on the terminal for this situation. The stack-cluster located behind portainer location 11, Stacks 101-110, has been included in Appendix 20 to give an impression of the rgv row length on the different lanes. Apart from some initial effects the rgv row only once exceeds the length of 2 rgv's. As rgv row of three are acceptable the stacks and three warehouse cranes seem to suffice.

Warehouse crane occupancy rates
The warehouse crane occupancy rates for the warehouse cranes located in Yard 1, 2 and 3 have been calculated for the simultaneous adjacent loading on all three berths. The occupancy rates of these warehouse cranes have been calculated in Appendix 20, resulting in:

Mean occupancy rate for a warehouse crane located in the 20' stack is 34.7 %.  
Mean occupancy rate for a warehouse crane located in the 40' stack is 23.2 %.

Mean occupancy rate for a warehouse crane located in the 20' stack is 43.4 %.  
Mean occupancy rate for a warehouse crane located in the 40' stack is 29.0 %.

These occupancy rates seem acceptable. It might even be possible to operate with fewer warehouse cranes.
8.5.2 Experiment 2

This simulation run concerns the adjacent loading of two feeder vessels moored at Berth 1 & 3 and the unloading of a mega vessel moored at Berth 2. This situation has been analysed in Appendix V.6 and has turned out to be the heaviest occupied situation for the lane, node and warehouse crane capacities on Yard 2. This is the reason that this situation has been chosen to be the normative situation on the terminal. In Chapter 8.6 some variations on the reference terminal will be carried out with this terminal configuration.

![Diagram showing the movement of portainers between berths](image)

*Figure 8.35, adjacent loading on Berth 1 & 3 and unloading on Berth 2*

**Portainer occupancy rates**
The occupancy rates of the portainers located on Berth 1 & 3 are comparable to the occupancy rates found in Experiment 1 of this chapter. The portainer occupancy rates become lower as the portainer is situated further away from Berth 2.
The portainer occupancy rates of the portainers located on Berth 2 show a similar picture as during the unloading without interference; Experiment 1 of Chapter 8.4. The occupancy rates are not as high, but the differences in odd- and even-numbered portainers are apparent again.

![Bar chart showing portainer occupancy rates](image)

*Figure 8.36, Portainer occupancy rates*
Handling production

The portainer occupancy rates as shown in Figure 8.36 result in the productions per berth and the vessel service times as shown in Table 8.5.

<table>
<thead>
<tr>
<th></th>
<th>Number of handled containers</th>
<th>Production per portainer (moves/hour)</th>
<th>Production per berth (moves/hour)</th>
<th>Production per berth (TEU/hour)</th>
<th>Service time (hours)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Berth 1</td>
<td>1050 (adj. load)</td>
<td>49.9</td>
<td>199.6</td>
<td>299.3</td>
<td>5.3</td>
</tr>
<tr>
<td>Berth 2</td>
<td>3150 (unload)</td>
<td>56.2</td>
<td>343.9</td>
<td>515.8</td>
<td>9.3</td>
</tr>
<tr>
<td>Berth 3</td>
<td>1050 (adj. load)</td>
<td>49.4</td>
<td>197.8</td>
<td>296.7</td>
<td>5.3</td>
</tr>
</tbody>
</table>

Table 8.7, berth productions

The service times for both adjacent loading feeder vessels are 0.2 hours longer compared to the undisturbed adjacent loading feeder vessel. The service time of the mega vessel is also 0.4 hours slower than the service times found during the undisturbed unloading.

Rgv waiting queues under the portainers

The length of the portainer queues is related to the portainer occupancy rates. The bigger the share of an empty queue in Figure 8.37 the lower the occupancy rates will be.

![Rgv waiting queues under the portainers on Berth 3](image)

Figure 8.37, rgv waiting rows under the portainers on Berth 1

Under the portainers of Berth 1 the rgv rows amount to 6 in length only once. This happens during the adjacent loading on Berth 1. The share of the empty rgv rows is about 20%. Increasing the numbers of rgv's on the terminal during adjacent loading to 12 might give more favourable results.

During the unloading of the mega vessel on Berth 2, the waiting row under the portainers occasionally grows to 6 rgv's in length. The share of an empty rgv row is about 7%.
The rgv waiting rows under the portainers located on Berth 3 are comparable to the waiting rows on Berth 1. This can be seen in Figure 8.39.

In Appendix 7 the rgv flows on the terminal have been calculated for this situation. From that calculation the nodes most liable to suffer long rgv rows have been identified. These are the nodes 110, 350, 108, 380 and 106 that connect the top inter-berth lanes with the yard-lanes. The nodes 508, 333, 510, 363 and 512 connect the bottom inter-berth lanes with the yard-lanes. The waiting rows of these nodes have been plotted in Appendix 21. What can be seen is the rgv waiting queues on the node seldom exceeds the length of 2 rgv’s.

The warehouse lanes of the warehouses located in Yard 2 are the busiest on the terminal for this situation. The stack cluster located behind portainer 11, Stacks 101-110, has been included in Appendix 21 to give an impression of the rgv row length on the different lanes. The rgv row on Warehouse lane 104 is located in a 20’ stack. The occupancy is high and at one point even exceeds the maximum length of 3 rgv’s as can be seen in Figure 8.40.
Warehouse crane occupancy rates
The warehouse crane occupancy rates for the warehouse cranes located in Yard 2 are the highest on the terminal during the simultaneous adjacent loading on all three berths. The occupancy rates of these warehouse cranes have been calculated in Appendix 21, resulting in:

Mean occupancy rate for a warehouse crane located in the 20’ stack is 70.1 %.  
Mean occupancy rate for a warehouse crane located in the 40’ stack is 47.3 %.

These occupancy rates are reasonably high which can also be concluded from the rgv rows on the warehouse lanes.
8.5.3 Experiment 3

This simulation run concerns the unloading of the two feeder vessels moored at Berth 1 & 3 and the adjacent loading of a mega vessel moored at Berth 2. This configuration provides the highest occupancy of the yards situated behind the feeder vessel berths.

Portainer occupancy rates
The portainer occupancy rates of both feeder vessels are quite recognisable. The portainer occupancy rates of the odd numbered portainers are slightly higher than those of the even numbered portainers. This is due to the relative longer driving cycles of the even numbered portainers. The portainers located on Berth 2 have a mean occupancy rate of about 83%. This can be increased by assigning more rgv’s to these portainers.

Figure 8.41, portainer occupancy rates

125
Handling production

Table 8.8 shows the handling production that can be achieved with this configuration. Despite the fact that the rgv’s from Berth 2 retrieve their containers from Yard 1 & 3, both feeder vessel service times are not influenced. The undisturbed service time for the unloading of a feeder vessel also equals 4.5 hours.

<table>
<thead>
<tr>
<th></th>
<th>Number of handled containers</th>
<th>Production per portainer (moves/hour)</th>
<th>Production per berth (moves/hour)</th>
<th>Production per berth (TEU/hour)</th>
<th>Service time (hours)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Berth 1</td>
<td>1050 (unload)</td>
<td>57.8</td>
<td>231.4</td>
<td>347.1</td>
<td>4.5</td>
</tr>
<tr>
<td>Berth 2</td>
<td>3150 (adj. load)</td>
<td>50.1</td>
<td>300.8</td>
<td>451.2</td>
<td>10.5</td>
</tr>
<tr>
<td>Berth 3</td>
<td>1050 (unload)</td>
<td>57.7</td>
<td>231.0</td>
<td>346.5</td>
<td>4.5</td>
</tr>
</tbody>
</table>

Table 8.8, berth productions

The adjacent loading of a mega vessel with containers being retrieved from Yard 1 & 3 takes about 10.5 hours. This is 0.8 hours more that the undisturbed adjacent loading of a mega vessel which takes 9.7 hours.

Rgv waiting queues under the portainers

The rgv rows located under the portainers on Berth 1 have not exceeded the length of 4 rgv’s during this simulation run. The small share of empty rgv rows under the portainers agrees with the high occupancy rates of the portainers.

![Rgv waiting queues under the portainers on Berth 3](image)

Figure 8.42, rgv waiting rows under the portainers on Berth 1

Figure 8.43 illustrates the length of the rgv waiting rows under the portainers located under the portainers on Berth 2. The empty row share is considerably higher compared to the rgv rows on Berth 1. The length of the rgv waiting row has amounted to 6 during this simulation run.
Rgv waiting rows at the nodes
The nodes most liable to have substantial rgv waiting rows are the access and exit nodes to the yard-lanes of Berth 1 & 3. They are the nodes 114, 243, 260, 273, 290, 504 and 506. These nodes have been joined to Appendix 22. The rgv rows on these nodes have never reached the length of 4 rgv’s during this simulation run.

Rgv waiting rows on the warehouse lanes
The warehouse lanes of the warehouses located in Yards 1 & 3 that are used by the unloading rgv’s have the highest occupancy. The stack-cluster containing the warehouse lanes 31-40 has been joined to Appendix 22. None of the rgv rows on the warehouse lanes on Berth 1 & 3 has exceeded the maximum length of 3, except warehouse lane 33.

Warehouse crane occupancy rates
The warehouse crane occupancy rates for the warehouse cranes located in Yard 1 & Yard 3 are the highest during this simulation run. They have been calculated in Appendix 22.

Mean occupancy rate for a warehouse crane located in the 20’ stack is 65.6 %.
Mean occupancy rate for a warehouse crane located in the 40’ stack is 43.7 %.

These occupancy rates are reasonably high which can also be concluded from the rgv rows on the warehouse lanes.
8.5.4 Experiment 4

In this experiment all the rgv's will be shuffling containers from their 'home' berth to an adjacent berth. This situation may occur when container vessels, having unloaded their containers in the yard behind their quays, have just left the terminal. During shuffling the rgv's are not subscribed to the portainers anymore but to the stack-clusters located behind the portainers. Per stack-cluster 10 rgv's will be active. Considering that shuffling will take place on all three the berths 10 * 14 stack-clusters = 140 rgv's will be active. This is the same amount of rgv's as during adjacent loading on three berths. As the portainers are not active during shuffling they will not need to be analysed.

![Diagram](attachment:image.png)

Figure 8.44, shuffling from all berths

Shuffle production

The shuffle production of the terminal during the shuffling on three berths will be defined as the total amount of shuffled containers per time unit. The total amount of shuffled containers is:

- Number of shuffled containers from Yard 1 to Yard 2: 1200
- Number of shuffled containers from Yard 2 to Yard 1: 1600
- Number of shuffled containers from Yard 2 to Yard 3: 1600
- Number of shuffled containers from Yard 3 to Yard 1: 1200

Total amount of shuffled containers on the terminal: 5600

The time necessary to shuffle this amount of containers is 23400 seconds or 6.5 hours.

This results in a total shuffle production of: \[ \frac{5600}{6.5} = 861.5 \] containers per hour, or 14.4 containers per minute.

Rgv waiting queues at the nodes

The nodes with the expected highest rgv rows have been analysed in order to get an impression on the busy ness on the terminal during the shuffling. These nodes are the same ones as have been analysed during the adjacent loading on all three berths, Experiment 1. The nodes with the highest expected rgv rows are nodes 110, 350, 108, 106 and 508, 333, 510, 363 and 512. The waiting rows of these nodes have been plotted in Appendix 23. What can be seen is that the lengths of the rgv queues occasionally exceed the length of 3 rgv's, but never the length of 4 rgv's. The node with the longest rgv queue is node 380 as can be seen in Figure 8.45.
**Waiting rows in the warehouse lanes**

The warehouse lanes of the warehouses in Yard 2 are the ones that will be visited most frequently during the simulation. Due to the high occupancy rates of the warehouse cranes, the rgv queues on the warehouse lanes may become too long. As an example, the rgv queue on warehouse lane 101 has been plotted in Figure 8.46.

**Warehouse crane occupancy rates**

The warehouse crane occupancy rates for the warehouse cranes located in Yard 2 are the highest on the terminal during shuffling on all three the berths. The occupancy rates of these warehouse cranes have been calculated in Appendix 23, resulting in:

Mean occupancy rate for a warehouse crane located in the 20’ stack in Yard 2 is 79.8 %
Mean occupancy rate for a warehouse crane located in the 20’ stack in Yard 2 is 53.2 %

These occupancy rates are extremely high, especially those of the 20’ stacks and its warehouse cranes. This, of course, is the main cause that the lengths of the waiting rows on the warehouse lanes are large. The reason that the warehouse crane occupancy rates are this high is because the numbers of shuffling rgv’s per stack-cluster is too high. By decreasing the number of shuffle rgv’s per stack-cluster, the occupancy rate of the warehouse cranes and the resulting rgv waiting rows will become less. The time needed to shuffle the same amount of containers will increase by appointing less shuffle rgv’s.

Bearing in mind the relative small differences between the adjacent loading and the unloading of a feeder or mega vessel and the extra amount of moves needed to shuffle the containers to the yards from where they can be fast loaded, shuffling will not be that profitable. The main reason is that the driving distances between the Yards are not far enough. Shuffling will therefore not be recommended.
### 8.5.5 Recapitulation on the simulation runs with 3 occupied berths

**Handling rates**
The service times performances found in Experiment 1-3 during the simultaneous handling of 3 container vessels have been plotted in Table 8.9.

<table>
<thead>
<tr>
<th></th>
<th>Feeder vessel moored at Berth 1 or 3</th>
<th>Mega vessel moored at Berth 2</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Unloading [hours]</td>
<td>Adjacent loading [hours]</td>
</tr>
<tr>
<td>Experiment 1</td>
<td>--</td>
<td>5.4</td>
</tr>
<tr>
<td>Experiment 2</td>
<td>--</td>
<td>5.3</td>
</tr>
<tr>
<td>Experiment 3</td>
<td>4.5</td>
<td>--</td>
</tr>
</tbody>
</table>

*Table 8.9, handling rates during simulation on 3 berths*

From the handling times displayed in Table 8.9, the service times of a feeder vessel and a mega vessel during high occupation on the reference terminal can be established. For a feeder vessel the maximum service time with 3 vessels moored vessels at the terminal is \(5.4 + 4.5 = 9.9\) hours. For a mega vessel moored at Berth 2 the same approach can be made. The maximum service time on the reference terminal will amount to \(9.3 + 10.5 = 19.8\) hours.

In Table 8.10 a comparison is made between the undisturbed service times and the maximum service times on the reference terminal.

<table>
<thead>
<tr>
<th></th>
<th>Undisturbed service times [hours]</th>
<th>Maximum service times [hours]</th>
<th>Difference [hours]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Unloading</td>
<td>Adjacent loading</td>
<td>Total</td>
</tr>
<tr>
<td>Feeder vessel</td>
<td>4.5</td>
<td>5.1</td>
<td>9.6</td>
</tr>
<tr>
<td>Mega vessel</td>
<td>8.9</td>
<td>9.7</td>
<td>18.6</td>
</tr>
</tbody>
</table>

*Table 8.10, comparison between the undisturbed and maximum service times*

What can be interpreted from Table 8.10 is that the differences between the undisturbed service times and the maximum service times on the reference terminal are not excessive. For a feeder vessel the difference as a percentage of the maximum service times is 3.3 %, while for the mega vessel it is 6.1 %.

These results defend the proposition to exclude the shuffling from the handling procedures on the terminal. The most important reason that the shuffling is not practicable for the reference terminal is the fact that the driving distances between the 3 yards is relatively small due to the compactness of the terminal. Retrieving a container from an adjacent stack during the loading of a vessel will not require immense driving distances compared to conventional terminals. Another very important issue is the fact that the containers are stacked in warehouses. This means that the amounts of shuffling moves in the yard itself are also strongly reduced. On a conventional terminal container stacks are pre-shuffled in such a way that the first containers to be loaded onto the vessel are easy reachable.
Operational rgv’s - portainer occupancies

The amount of operational rgv’s on the reference terminal during the simulation runs with 3 occupied berths has been established to be 6 rgv’s per portainer during the unloading, and 10 per portainer during the adjacent loading of a container vessel. As the portainer occupancy rates during the adjacent loading of a feeder or mega vessel are relatively low increasing the number of rgv’s to 12 per portainer might be a solution. Especially considering the fact that the waiting rows on the nodes and warehouse lanes are not very long. A simulation run with 12 rgv’s has been carried out in the next chapter. The simulation run is based on the situation as it exists in Experiment 2.

Warehouse crane occupancy rates

The warehouse crane occupancy rates on the reference terminal with 3 occupied berths have been included to Table 8.11.

<table>
<thead>
<tr>
<th>Warehouse crane occupancy rates with 3 occupied berths (%)</th>
<th>Yard 1</th>
<th></th>
<th>Yard 2</th>
<th></th>
<th>Yard 3</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>20' stack</td>
<td>40' stack</td>
<td>20' stack</td>
<td>40' stack</td>
<td>20' stack</td>
<td>40' stack</td>
</tr>
<tr>
<td>Experiment 1</td>
<td>43.4</td>
<td>29.0</td>
<td>34.7</td>
<td>23.2</td>
<td>43.4</td>
<td>29.0</td>
</tr>
<tr>
<td>Experiment 2</td>
<td>--</td>
<td>--</td>
<td>70.1</td>
<td>47.3</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>Experiment 3</td>
<td>65.6</td>
<td>43.7</td>
<td>--</td>
<td>--</td>
<td>65.6</td>
<td>43.7</td>
</tr>
</tbody>
</table>

Table 8.11, warehouse crane occupancy rates

With the data from this table the necessary number of warehouse cranes can be determined. This has been done below.

First the normative situation must be determined from Experiment 1-3 by establishing the mean warehouse crane occupancy rate over all the yards.

The 20' container stacks:

Exp. 1: \[
\frac{43.4*4 + 34.7*6 + 43.4*4}{14} = 40\%
\]

Exp. 2: \[
\frac{0*4 + 70.1*6 + 0*4}{14} = 30\%
\]

Exp. 3: \[
\frac{65.6*4 + 0*6 + 65.6*4}{14} = 37.5\%
\]

The occupancy rate is the normative for Experiment 1. The warehouse cranes in the 20' stacks have a mean occupancy rate of 40%. The target occupancy rate for this situation is 70 %, resulting in 16 warehouse cranes in the 20' stacks.

The same calculation has been made for the 40' stack:

Exp. 1: \[
\frac{29.0*4 + 23.2*6 + 29.0*4}{14} = 26.5\%
\]

Exp. 2: \[
\frac{0*4 + 6*47.3 + 0*4}{14} = 20.3\%
\]
Exp. 3: \( \frac{43.7 \times 4 + 0 \times 6 + 43.7 \times 4}{14} = 25\% \)

The occupancy rate is the normative for Experiment 1. The warehouse cranes in the 40' stacks have a mean occupancy rate of 26.5%. The target occupancy rate for this situation is 70%, resulting in 16 warehouse cranes in the 40' stacks. As there are three warehouse rows present in the 40' yards the number of warehouse cranes has been determined to amount to 18.

It has to be noted that during all simulation runs on the reference terminal the warehouse crane handling cycles have not been variable but deterministic. Variable operating warehouse cranes will probably have slightly different occupancy rates. For this reason a simulation run has been carried out with variable handling cycles. This simulation run is based on the situation in Experiment 2 as it has the highest occupation rates in the yard.

**Rgv waiting queues on the nodes**
The normative situation for determining the needed rgv waiting row space between 2 inter-berth lanes is the adjacent loading on all 3 berths. In this situation the inter-berth lanes will have to digest the largest amount of rgv's. The required space for these waiting rows has been based on the row lengths located between the crossings of inter-berth lanes with the extensions of the yard-lanes. Figure 8.47 provides an illustration where the locations of the waiting places are represented by the black dots.

![Diagram showing rgv waiting spaces and yard-lanes]

**Figure 8.47, a part of the top-view of the layout of the reference terminal**

The simulation run has led to the following maximum row lengths.

**Top inter-berth lanes.**
- Maximum row length that has occurred between inter-berth lane 1 and 2: 3 rgv's
- Maximum row length that has occurred between inter-berth lane 2 and 3: 3 rgv's
- Maximum row length that has occurred between inter-berth lane 3 and 4: 5 rgv's
- Maximum row length that has occurred between inter-berth lane 4 and yard-lanes: 5 rgv's

**Bottom inter-berth lanes.**
- Maximum row length that has occurred between inter-berth lane 8 and 7: 3 rgv's
- Maximum row length that has occurred between inter-berth lane 7 and 6: 4 rgv's
- Maximum row length that has occurred between inter-berth lane 6 and 5: 4 rgv’s
- Maximum row length that has occurred between inter-berth lane 5 and yard-lanes: 4 rgv’s

These maximum row lengths will occasionally occur during adjacent loading on all 3 berths. In the next chapter these results will be discussed and integrated in the layout of the terminal.
8.6 Simulation runs of variations on the reference terminal

In the last chapters a variety of runs have been carried out based on the reference terminal. In this chapter some alterations to the reference terminal have been simulated and analysed. These alternations have been applied to the normative situation on the terminal. The established normative situation is the adjacent loading of the feeder vessels moored at Berth 1 & 3 combined with the unloading of the mega vessel moored at Berth 2. This is Experiment 2 of Chapter 8.5.

The experiments and their alternations to the reference terminal can be seen in Table 8.12.

<table>
<thead>
<tr>
<th>Experiment</th>
<th>Berth 1</th>
<th>Berth 2</th>
<th>Berth 3</th>
<th>Alternation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Adjacent loading</td>
<td>Unloading</td>
<td>Adjacent loading</td>
<td>From 10 to 12 rgv’s per portainer on the adjacent loading berths</td>
</tr>
<tr>
<td>2.</td>
<td>Adjacent loading</td>
<td>Unloading</td>
<td>Adjacent loading</td>
<td>The addition of disturbances to the portainer handling cycles</td>
</tr>
<tr>
<td>3.</td>
<td>Adjacent loading</td>
<td>Unloading</td>
<td>Adjacent loading</td>
<td>Crane handling times have been made variable instead of deterministic</td>
</tr>
</tbody>
</table>

*Table 8.12, description of alternations to the reference terminal*
8.6.1 Increasing the number of adjacent loading rgv's to 12

During this simulation run the number of active rgv's working for the portainers located on the feeder vessel Berths 1 & 3 have been increased to 12. The most important results of this alternation will be dealt with in this chapter.

**Portainer occupancy rates**
As can be seen in Figure 8.47 the portainer occupancy rates of the portainers located on Berth 1 & 3 have increased with about 10 % compared to the reference terminal.

![Portainer occupancy rates graph](image)

*Figure 8.48, portainer occupancy rates*

**Handling production**
The handling production on the three berths can be seen in Table 8.13.

<table>
<thead>
<tr>
<th>Berth</th>
<th>Number of adjacent loaded containers</th>
<th>Production per portainer (moves/hour)</th>
<th>Production per berth (moves/hour)</th>
<th>Production per berth (TEU/hour)</th>
<th>Service times (hours)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1050 (adj. load)</td>
<td>55.9</td>
<td>223.8</td>
<td>335.7</td>
<td>4.7</td>
</tr>
<tr>
<td>2</td>
<td>3150 (unload)</td>
<td>54.8</td>
<td>328.8</td>
<td>493.2</td>
<td>9.6</td>
</tr>
<tr>
<td>3</td>
<td>1050 (adj. load)</td>
<td>56.1</td>
<td>224.5</td>
<td>336.8</td>
<td>4.7</td>
</tr>
</tbody>
</table>

*Table 8.13, handling productions with 12 rgv's*

Compared to the similar run with 10 rgv's the service times on Berth 1 & 3 have decreased to under the service times of the undisturbed adjacent loading of a feeder vessel. The service times no Berth 2 have increased a bit. The increase of the service time on Berth 2 is due to the fact that that there are more rgv's on the lanes and in the stacks causing some delay.

**Rgv waiting queues under the portainers**
Compared to the similar run with 10 rgv's, the percentage of empty waiting rows under the portainers on Berth 1 & 3 have become smaller. This can be seen in the following figures.

**Figure 8.49, rgv waiting rows under the portainers on Berth 1**

**Figure 8.50, rgv waiting rows under the portainers on Berth 2**

**Figure 8.51, rgv waiting rows under the portainers on Berth 3**
Rgv waiting queues on the nodes and on the warehouse lanes
In Appendix VIII.13 the rgv waiting queues on the nodes and warehouse lanes can be found.
They pose no threats to the terminal.
8.6.2 The loading and unloading of a vessel using different portainer cycle times

Experiment
The portainer cycle times used at the reference terminal are the shortest cycle times possible. In reality these cycle times will not be this perfect. Because that would mean the portainer driver will not make any mistakes that would decrease the cycle times of the portainer. Therefore some extra time penalties have been added to model to be interpreted as disturbances. Two kinds of disturbances are added.

A small disturbance with a maximum of 5 seconds and a distribution function as shown in Figure 8.52.

![Figure 8.52, distribution of a small disturbance in the portainer handling time](image)

A big disturbance that only occurs once in a hundred portainer moves, with a mean of 300 [s] as shown in Figure 8.53.

![Figure 8.53, distribution of a big disturbance in the portainer handling time](image)

Including these two disturbances, the new mean portainer cycle time is 63 seconds.
The portainer cycle times

The mean portainer cycle times of the basic run are shown in Table 8.14 where they are compared with the cycle times of the experiment.

<table>
<thead>
<tr>
<th>Portainer</th>
<th>Old portainer cycle time [s]</th>
<th>experiment portainer cycle time [s]</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>56.66</td>
<td>59.28</td>
</tr>
<tr>
<td>6</td>
<td>57.04</td>
<td>61.45</td>
</tr>
<tr>
<td>7</td>
<td>56.90</td>
<td>60.58</td>
</tr>
<tr>
<td>8</td>
<td>57.00</td>
<td>62.54</td>
</tr>
<tr>
<td>9</td>
<td>56.86</td>
<td>61.08</td>
</tr>
<tr>
<td>10</td>
<td>57.08</td>
<td>65.85</td>
</tr>
<tr>
<td>mean cycle time</td>
<td>56.92</td>
<td>61.80</td>
</tr>
</tbody>
</table>

Table 8.14, the mean portainer cycle times

In this table the difference between the old and the new mean cycle times becomes clear. The difference is 4.88 seconds.

Using the cycle times and the occupancy rates of the portainers, the productions per portainer can be calculated. It is clear that the portainer productions will be lower in the experiment while the mean cycle times are higher.

First the portainer production is considered without the portainer occupancy rates, the production is defined as in (8.1)

$$\text{production}\ [\text{moves/ min}] = 60 / \text{mean cycle time}\ [\text{sec/ move}] \quad (8.1)$$

The mean cycle times of the two runs as shown in table could suggest that the portainer productions in the experiment are 8.6 % lower than in the original situation (8.2).

$$\frac{61.8[\text{sec}]}{56.9[\text{sec}]} = 1.086 \rightarrow 8.6\% \quad (8.2)$$

Figure 8.53 shows the portainer productions from the run compared to the original situation. Relatively the productions are not as low as expected. This is because the portainer occupancy rates in the experiment are higher than in the original run.
In the case of Berth 2 the big disturbances do not disturb the unloading process at the terminal too much. Obviously they only result in longer queues under the portainer locations. With a maximum row length of 6 while there are only 6 rgv’s per portainer used during unloading. The queues under the portainers are shown in Appendix 25.

When a portainer is loading a vessel with containers from an adjacent berth, as is the case on Berth 1 & 3, a big disturbance could prove to be a bigger problem because more rgv’s per portainer are used in this process. Figure 8.56 shows a result of the experiment. At one point a queue of 8 rgv’s arises. This is the result of a big disturbance of 500 seconds. The queue length of 5 rgv’s in the picture is the result of a disturbance of 200 seconds.
Portainer 3 queue

Figure 8.56, rgv queue under portainer 3
This second disturbance is shown in together with the rgv queue which is created by the disturbance.

Figure 8.57, reaction to a big disturbance

The line shows the disturbance in the cycle times and the columns show the rgv waiting queues under portainer 3.
8.6.3 Stacking crane cycle times

The crane cycle times used at the reference terminal have deterministic cycle times of 80 seconds. In reality these cycle times will be different depending on the location of the containers which have to be retrieved or stacked. These cycle times can differ from 44 seconds for a container to be retrieved from the first floor up to 116 seconds for a container that has to be delivered to the top of a stack.

This is a reason to experiment with a triangular function for the crane cycle times. This will be a triangular function with a minimum of 44 [s], a maximum of 116 [s] and a mean of 80 seconds. The results of this run will be compared with the results of the reference terminal. Important performance parameters during this run are agv queues in the stacks and under the portainers and the portainer productivity.

Portainer productivity

The portainer occupancy rates in the second run are on the average 3.5% lower and the mean portainer cycle time is the same. This means the production will be 3.5% less. The portainer occupancy rates are shown in Figure 8.58, portainer occupancy rates of the experiment compared to the basic run.

![Portainer occupancy rates](image)

Figure 8.58, portainer occupancy rates of the experiment compared to the basic run

The productions for the portainers are shown in Table 8.12.

<table>
<thead>
<tr>
<th>Berth</th>
<th>Number of containers</th>
<th>Production per portainer (moves/hour)</th>
<th>Production per portainer (moves/hour)</th>
<th>Difference in handling time [hours]</th>
<th>Difference in handling time [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1050 loading</td>
<td>49.9</td>
<td>47.5</td>
<td>0.2</td>
<td>4.8</td>
</tr>
<tr>
<td>2</td>
<td>3150 unloading</td>
<td>57.3</td>
<td>55.7</td>
<td>0.2</td>
<td>2.8</td>
</tr>
<tr>
<td>3</td>
<td>1050 loading</td>
<td>49.4</td>
<td>47.3</td>
<td>0.2</td>
<td>4.3</td>
</tr>
</tbody>
</table>

Table 8.15, portainer productions during the basic run and the crane experiment
The difference is about 12 minutes at each berth.

Rgv waiting queues on the warehouse lanes
The warehouse lanes in the 20' warehouses in Yard 2 are the busiest of the terminal. These queues sometimes exceed the length of 3 rgv's. There is even a case where the queue length exceeds the length of 4 rgv's, which is not acceptable. This queue is shown in Figure 8.59.

Figure 8.59, the rgv queue on warehouse lane 82

The queues in the 40' stacks never exceed the length of 3 rgv's. The queues on the warehouse lanes are shown in appendix 26.
9 Discussion of the Simulation Results

The results of the simulation runs will be discussed in this chapter. Adding some critical notes will help to see the optimistic results in some perspective.

9.1 Introduction

The results of this study are the outcome of strongly schematised simulation runs. The parts of the simulation runs that have been most strongly schematised are the portainer handling rates and the warehouse crane handling rates. Assumptions concerning the properties of the rgv's and the stacking height in the warehouses have to be regarded with caution. Each of these items will be described here.

Portainer handling rates
As has been illustrated in Chapter 4 the portainer handling rate varies with the location of the container in the hold of a vessel. The variable handling rates have been taken from FAMAS [L.12]. Apart from these variations the simulation runs concerning the reference terminal have not been carried out with other irregularities. Some of the most important irregularities that have been left out of the simulation are:

- Time loss due to jammed containers in the hold of the vessel.
- Time loss due to the removing of the hatches from the vessels.
- Time loss due to the lateral movements of the portainers.

Apart from these time losses the portainers have been modelled in such a way that each portainer will unload more or less the same amount of containers. This schematisation has a positive effect on the service time of a vessel since in reality the containers are not distributed over the holds in such a favourable way.

Warehouse crane handling rates
The average handling cycle of a warehouse crane has been determined in Appendix 8. For a 13-storey warehouse the average time it takes for a warehouse crane to complete a handling cycle amounts to 80 seconds. This average handling rate is the deterministic value of the handling rate of a warehouse crane in the simulation model. This schematisation excludes the variations in handling rates that will take place in reality. Effects that might occur due to these variations, like extra rgv congestion resulting in time loss, have consequently been left out of the analysis.

Stacking height
The stacking height in the warehouses has been based on an assumption that it is possible to build a 13-storey warehouse and a warehouse crane able to handle containers over these heights. A study has to be carried out if such a container warehouse can be realised. If a 13-storey warehouse cannot be realised the consequence of this assumption is double. First, lower warehouses will result in more required land area, and second, the rgv delivery and retrieving cycles will demand longer transportation cycles. This means more rgv's have to be assigned per portainer if the vessel handling rates are to be kept constant.
Rgv properties
The properties of the rgv's have been assumed to a certain extend. The time required to
complete the 90° rotation of the wheels of the rgv during turning has been assumed to be 4
seconds. Furthermore the acceleration of the rgv's has been assumed to reach 1 m/s². If
these assumptions prove to be incorrect rgv cycles will require more time resulting in longer
service times for the container vessels or more rgv's per portainer.

The results obtained from the simulation runs may thus be optimistic. In order to obtain more
lifelike and realistic results the above-discussed issues should be taken into account. That
makes the character of this study a more orientating and academic one. These results can
not be expected to be matched in reality.

Nevertheless, the results are encouraging and will be discussed in the next parts of this
chapter. The outcome of the simulation runs on the reference terminal must be regarded as a
first simulation round. By adding disturbances to the handling cycles of the portainers and
warehouse cranes, the influences of these disturbances can be compared to the simulation
runs of the reference terminal used in this study.

9.2 Interpretation of the simulation results

In this paragraph the performance indicators of the simulation runs as dealt with in Chapter 8
will be interpreted and discussed.

9.2.1 Handling performance

The handling performances resulting in the vessel service times for both feeder vessels and
mega vessels have been illustrated below.

The displayed results contain the handling rates and service times during the unloading and
the adjacent loading procedure. The reduction on the service time of a container vessel
gained by the shuffling procedure is not considerable compared to the time needed to
complete the adjacent loading procedure. Especially if the number of adjacent loading rgv's
per portainer is increased to 12 rgv's per portainer.

The shuffling procedure will therefore not be taken into account any further. Table 9.1 gives
an overview of the simulation results.

<table>
<thead>
<tr>
<th></th>
<th>Feeder Vessel</th>
<th></th>
<th>Mega Vessel</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Handling rates</td>
<td>Service times</td>
<td>Handling rates</td>
<td>Service times</td>
</tr>
<tr>
<td></td>
<td>(mvs./hour)</td>
<td>(hours)</td>
<td>(mvs./hour)</td>
<td>(hours)</td>
</tr>
<tr>
<td></td>
<td>Unl Adj Average</td>
<td>Unl Adj Total</td>
<td>Unl Adj Average</td>
<td>Unl Adj Total</td>
</tr>
<tr>
<td>Undisturbed</td>
<td>236 206 221</td>
<td>4.5 5.1 9.6</td>
<td>352 324 338</td>
<td>8.9 9.7 18.6</td>
</tr>
<tr>
<td>Disturbed</td>
<td>231 194 213</td>
<td>4.5 5.4 9.9</td>
<td>344 300 322</td>
<td>9.2 10.5 19.8</td>
</tr>
</tbody>
</table>

*Table 9.1, the handling rates and service times*
Undisturbed unloading procedure

Compared to the preliminary calculation made in Chapter 3.6 the handling rates of the undisturbed simulation runs during unloading do not differ greatly for both the feeder vessels and mega vessels. This can be seen in Table 9.2.

<table>
<thead>
<tr>
<th>Unloading procedure</th>
<th>Feeder vessel</th>
<th>Mega vessel</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preliminary calculation</td>
<td>240 mvs/h</td>
<td>360 mvs/h</td>
</tr>
<tr>
<td>Undisturbed simulation</td>
<td>236 mvs/h</td>
<td>352 mvs/h</td>
</tr>
</tbody>
</table>

Table 9.2, comparison between undisturbed situation and the ideal preliminary calculation

The main reason that these deviations are relatively so small is because the simulation runs have been carried out without any great disturbances in the portainer cycles or in the warehouse crane cycles. The undisturbed simulation runs can therefore be regarded as ideal runs. An important conclusion that can be drawn however is the fact that the horizontal transportation turns out not to be a bottleneck when the portainers and warehouse cranes are operating at full capacity. The amount of 6 rgv’s operating for a portainer during the unloading procedure is satisfactory as well.

It is recommended to include handling cycle irregularities to the simulation program and compare the outcome to the results of the reference terminal that has been used in this study.

Undisturbed adjacent loading procedure

The simulation runs concerning the adjacent loading procedure show a greater deviation compared to the preliminary calculation as can be seen in Table 9.3.

<table>
<thead>
<tr>
<th>Adjacent loading procedure</th>
<th>Feeder vessel</th>
<th>Mega vessel</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preliminary calculation</td>
<td>240 mvs/h</td>
<td>360 mvs/h</td>
</tr>
<tr>
<td>Undisturbed simulation</td>
<td>206 mvs/h</td>
<td>324 mvs/h</td>
</tr>
</tbody>
</table>

Table 9.3, comparison between undisturbed situation and the ideal preliminary calculation

The handling performances on the terminal during adjacent loading are relatively somewhat lower than during the unloading procedure. As was the case during the unloading procedure great disturbances have not been included to the handling rates of the portainers and the warehouse cranes. The main reason that the handling rates are relatively lower than during the unloading procedure is because the occupancy rates of the portainers are lower than during unloading caused by operating with 10 rgv’s per portainer during the adjacent loading procedure. This is a result. The adjacent loading simulation experiment in Paragraph 8.6.1 with 12 rgv’s per portainer confirms this conclusion as the handling rates increased.
A recommendation is simulating the adjacent loading procedure with more than 10, preferably 12 rgv’s, and including random disturbances in both portainer and warehouse crane handling cycles.

**Disturbed unloading procedure**

By carrying out simulation runs handling with than one container vessel the occupancy of the terminal has been increased. These runs are called the disturbed runs as opposed to the undisturbed runs which are carried out with only one moored vessel. As is the case during the undisturbed unloading procedure the handling rates are still relatively high. This can be seen in Table 9.4.

<table>
<thead>
<tr>
<th>Unloading procedure</th>
<th>Feeder vessel</th>
<th>Mega vessel</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preliminary calculation</td>
<td>240 mvs/h</td>
<td>360 mvs/h</td>
</tr>
<tr>
<td>Disturbed simulation</td>
<td>231 mvs/h</td>
<td>344 mvs/h</td>
</tr>
</tbody>
</table>

*Table 9.4, comparison between disturbed situation and the ideal preliminary calculation*

As is the case with the undisturbed situations the handling cycles of the portainers and the warehouse cranes have not been penalised with random disturbances. The reason for the decrease of the handling production therefore is the increased numbers of rgv’s operating on the different berths. More operating rgv’s means a higher chance for a rgv to wait on lane crossings.

It is recommended to include handling cycle irregularities to the simulation program and compare the outcome to the results of the reference terminal that has been used in this study.

**Disturbed adjacent loading procedure**

The simulation runs concerning the adjacent loading procedure show a greater deviation compared to the preliminary calculation as can be seen in Table 9.5.

<table>
<thead>
<tr>
<th>Adjacent loading procedure</th>
<th>Feeder vessel</th>
<th>Mega vessel</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preliminary calculation</td>
<td>240 mvs/h</td>
<td>360 mvs/h</td>
</tr>
<tr>
<td>Disturbed simulation</td>
<td>194 mvs/h</td>
<td>300 mvs/h</td>
</tr>
</tbody>
</table>

*Table 9.5, comparison between disturbed situation and the ideal preliminary calculation*

Comparison between the results of the simulation runs of the undisturbed and the disturbed variants of adjacent loading shows a decrease of production of a few percentages. The reason for this decrease is the increase of operational rgv’s on the terminal. Increasing the number of rgv’s per portainer will further increase the occupancy rates of the portainers and thus the handling capacity per berth.
A recommendation is to simulate the adjacent loading procedure with more than 10, preferably 12 rgv's, and to include random disturbances in both portainer and warehouse crane handling cycles. These recommendations have been carried out in separate simulation runs and will be discussed in the next paragraph.

9.2.2 Influence of variations to the reference terminal

Three experiments have been conducted in Paragraph 8.6 concerning variations to the reference terminal. One experiment concerning the adjacent loading with 12 rgv's per portainer, one with variable crane handling cycles and one with the addition of random disturbances to the portainer handling cycles. All these experiments have been conducted to the normative situation on the terminal. Integrating all three these experiments into 1 simulation run will show the amplifying effects they have on each other and is therefore recommended.

Simulation with 12 rgv’s during adjacent loading
What stands out in this discussion is the fact that the horizontal transportation operating on the terminal has a high capacity, especially during unloading. If the amount of operational rgv’s per portainer is increased to 12 during an adjacent loading procedure the portainers will achieve occupancy rates comparable to the occupancy rates achieved during the unloading procedure. This means that the horizontal transportation units, the rgv’s, are able to cope with the high production rates of the portainers. Simulation experiment 1 in Chapter 8.6 illustrates this.

Simulation with additional disturbances to the portainer handling cycles
The added disturbances to the portainer handling cycles have an influence on the production capacity of the terminal. The higher the mean handling cycles will be due to these disturbances the lower the mean handling production will be.

An interesting result however is the recovery rate of the rgv waiting queue under the portainers. Logically, the rgv waiting queue will increase as a portainer handling cycle is penalised with a long disturbance. The recovery rate is rather high as the rgv waiting queue slinks to normal proportions after a limited period of time. This can be seen in Figure 8.55 and 8. 56. This result shows that the horizontal transportation can be regarded as flexible.

Simulation with variable warehouse crane handling cycle times
Introducing a triangular warehouse crane cycle function has a negative effect on the container vessel service time, because these irregularities create additional waiting times for the rgv’s on the warehouse lanes. The waiting queues will grow longer if for instance a warehouse crane receives a number of longer handling assignments following each other, as can be seen in Figure 8.58 in Chapter 8.
The time loss caused by this experiment has been quantified and shown in Table 9.6.

<table>
<thead>
<tr>
<th>Number of containers</th>
<th>Basic reference terminal simulation run</th>
<th>Variable crane handling cycle simulation run</th>
<th>Difference in handling time</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Production per container (mvs/hour)</td>
<td>Production per berth (mvs/hour)</td>
<td>Handling time (hours)</td>
</tr>
<tr>
<td>Berth 1</td>
<td>49.9</td>
<td>199.6</td>
<td>5.3</td>
</tr>
<tr>
<td>Berth 2</td>
<td>57.3</td>
<td>343.8</td>
<td>9.2</td>
</tr>
<tr>
<td>Berth 3</td>
<td>49.4</td>
<td>197.8</td>
<td>5.3</td>
</tr>
</tbody>
</table>

Table 9.6, portainer productions of the basic run and the crane experiment

The introduction of variable handling cycles for the warehouse cranes seems not to influence the service times to a dramatic extent. The adjacent loading time will grow with about 4.5 % on Berth 1 & 3, while the unloading procedure on Berth 2 will grow to about 2.8 % on Berth 2. It must be noted however that although the simulated situation on the terminal is a heavy occupied one, more runs should be carried out in order to obtain a proper understanding on the influence of this variation to the reference terminal.

9.2.3 Crane occupancy rates

The normative situation from which the occupancy rates of the warehouse cranes are the highest is the adjacent loading on all 3 berths as is stated in Paragraph 8.5.5. In Table 9.7 these crane occupancy rates together with the number of cranes that are active on the reference terminal have been plotted.

<table>
<thead>
<tr>
<th></th>
<th>Yard 1</th>
<th>Yard 2</th>
<th>Yard 3</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>20' stack</td>
<td>40' stack</td>
<td>20' stack</td>
</tr>
<tr>
<td>Occupancy rate (%)</td>
<td>43.3</td>
<td>29.0</td>
<td>34.7</td>
</tr>
<tr>
<td>Number of warehouse cranes</td>
<td>8</td>
<td>12</td>
<td>12</td>
</tr>
</tbody>
</table>

Table 9.7, number of cranes and their occupancy rates

As the warehouse cranes are able to transfer through the warehouse rows, they can be concentrated to locations where the demand for production is highest. The transfer possibility increases the flexibility of the container storage system and offers a possibility to make more efficient use of the warehouse cranes. The occupancy rates of the warehouse cranes on the reference terminal are rather low as can be seen in Table 9.7.

Assuming that a warehouse crane occupancy rate of 70 % is acceptable allows determining the number of needed warehouse cranes. This has been calculated in Paragraph 8.5.5. The amount of warehouse cranes operating in 20' container stacks is 16. As the 20' container stacks consist of 2 warehouse rows, each of these rows will host 8 warehouse cranes.

The 40' container stacks will host 18 warehouse cranes. This amount of cranes will be spread out over 3 warehouse rows resulting in 6 warehouse cranes per row.

It is recommended to carry out simulation runs with the established amount of cranes.
9.2.4 Consequences for the terminal area

The amount of land area that the reference terminal occupies is based on a terminal layout in which the rgv waiting rows on the crossings have not been included. By carrying out an analysis on the waiting row length for a number of nodes the required space can be determined.

The required land area for the reference terminal is $178 \times (1000-132) \approx 15.5$ ha as can be seen in Appendix 1. The required rgv parking space behind the yards is not included.

9.2.5 Integration of rgv waiting rows on the terminal

The land usage of the final terminal in this study is the land usage needed for the simulated reference terminal plus the addition of space needed for the rgv’s waiting rows on the nodes. From the rgv waiting row lengths, included in the appendices, an impression can be obtained for the needed space per node. The normative situation of the reference terminal for the node rows is the adjacent loading on all 3 berths. This is Experiment 1 dealt with in Paragraph 8.5.1.

The maximum required rgv waiting row length between 2 inter-berth lanes, has been plotted in Table 9.8.

<table>
<thead>
<tr>
<th>Location of the waiting rgv row</th>
<th>Maximum waiting row length found during the simulation run (number of rgv's)</th>
<th>Applied waiting row length on the terminal (number of rgv's)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inter-berth lanes located at the top of the terminal</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>Between inter-berth lanes 1 &amp; 2</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>Between inter-berth lanes 2 &amp; 3</td>
<td>5</td>
<td>9</td>
</tr>
<tr>
<td>Between inter-berth lanes 3 &amp; 4</td>
<td>5</td>
<td>9</td>
</tr>
<tr>
<td>Between inter-berth lanes 4 &amp; yard</td>
<td>3</td>
<td>9</td>
</tr>
<tr>
<td>Inter-berth lanes located at the bottom of the terminal</td>
<td>4</td>
<td>9</td>
</tr>
<tr>
<td>Between inter-berth lanes yard &amp; 5</td>
<td>4</td>
<td>9</td>
</tr>
<tr>
<td>Between inter-berth lanes 5 &amp; 6</td>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>Between inter-berth lanes 6 &amp; 7</td>
<td>4</td>
<td>6</td>
</tr>
</tbody>
</table>

*Table 9.8, simulation results for the waiting rows and applied waiting row length*

The maximum waiting row length found during the normative simulation run does not occur very often as can be seen in Appendix 20. However, the fact that longer rgv waiting rows have not been found during this simulation run does not mean that the rows will never exceed the found lengths.

Because the rgv waiting row lengths have not been statistically analysed the applied rgv waiting row lengths have been chosen rather arbitrary. The chance that the rgv waiting rows on the reference terminal will exceed the applied permitted length is low. Introducing random disturbed operating times for the portainers and the warehouse cranes though, will probably lead to longer rgv waiting rows. It is therefore recommended that a simulation run should be
carried out during the adjacent loading on all 3 berths with random disturbed operating times for portainers and warehouse cranes.

If the rgv waiting row length exceeds the applied permitted rgv waiting row length nonetheless, the rgv will have to drive past the crossing point and request an alternate route. Another consideration to be made is the fact that the destinations of the rgv’s are completely randomly chosen. By introducing a more structured rgv destination choosing system potential high-density situations for rgv’s can be prevented.

The integration of the rgv waiting row space into the reference terminal has been shown in Figure 9.1. The available space between inter-berth lanes 1 and 2 and inter-berth lanes 2 and 3 is 6 rgv’s, while the available space between inter-berth lanes 3 and 4 and inter-berth lane 4 and the yard is 9 rgv’s. The same space has been reserved for the bottom inter-berth lanes. The complete top view of the terminal can be seen in Appendix 29.

![Figure 9.1, available waiting spaces between the inter-berth lanes](image)

By integrating the waiting spaces for the rgv’s into the reference terminal, the rgv travelling cycles will increase. This will probably not affect the results to much as the increase of travelling distance is relatively small.

The total of needed land area has increased to $185 \times 884 = 16.5$ ha. This is the area without rgv parking and service buildings.
9.3 Queuing theory

In order to determine the annual berth production of the terminal longer runs are required. This model is to detailed to simulate these longer periods. Using results of the reference terminal runs linked to vessel arrival schemes could say something over the turn around times of the vessels and the vessel queues in the anchorage over a longer period of time. The queuing system can be described by the inter arrival distribution of the vessels, the distribution of the service times and the number of berths. As this study concerns a none existing terminal some assumptions concerning the arrival times of the vessels must be made.
For the mega vessel berth and the two feeder berths two situations have been examined, using different arrival distribution functions.

9.3.1 Mega vessel berth

Concerning the mega vessel two situations using different arrival time and service time distributions are looked at.

M/D/1
- Negative exponential inter arrival time distribution
- Deterministic service time distribution
- 1 Berth

E_r/E/1
- Erlang distribution for the inter arrival times
- Erlang distribution for the service times
- 1 Berth

M/D/1
The negative inter arrival distribution (N.E.D.) is used to model inter arrival times when arrivals are completely random.
The service times are assumed to have no variability, that is \( \sigma^2 = 0 \), which means all the service times are assumed the constant value \( 1/\mu \).

If assumed that the service time of a mega vessel including the mooring is 22 hours and on the average 1 vessel per 36 hours arrives at the terminal, the occupancy rate, average waiting time and average waiting queue can be calculated.

\[ \rho = \frac{\lambda}{\mu} \]  \hspace{1cm} (9.1)

where:
\( \mu \) = service rate
\( \lambda \) = arrival rate

The occupancy rate in this case is \( (1/36)/(1/22) = 61\% \); The average number of customers in the waiting queue \( N_w \) can be calculated using 9.2.

\[ N_w = \frac{\rho^2}{2(1-\rho)} \]  \hspace{1cm} (9.2)

The average number of mega vessels in the queue is 0.48. The average waiting time for these vessels \( W \), can be calculated using formula 9.3;
\[ W = \frac{\lambda \mu^{-2}}{2(1 - \rho)} \quad (9.3) \]

The average waiting time per mega vessel 17.23 hours.
The average turn around time for a mega vessel in this system using these values will be 39.23 hours.

**E/\text{E/1}**

In reality the arrival pattern of the mega vessel will not be completely random. Neither will the service times be completely deterministic.

The distribution of the arrival times of the vessels at the terminal is assumed to have an average of 36 hours and a standard deviation of 12 hours. The service times of the vessels have an average of 22 hours and a standard deviation of 6 hours. The distributions can be assumed to have an Erlang distribution.

While the probability equations are not algebraically soluble the values of:
- utilisation (\(u\))
- variability of the \(E_k\) distribution of the inter arrival times (\(\nu_a=(s.d/\text{mean})^2\))
- variability of the \(E_k\) distribution of the service times (\(\nu_s=(s.d/\text{mean})^2\))

can be obtained from tables.

The utilisation of the berth is 61%.

The coefficient of variation of the arrival time;

\[ \frac{\text{s.d}}{\text{mean}} = \frac{12}{36} = 0.33 \quad (9.4) \]

so \(\nu_a = (c.v)^2\) of arrival time = \((0.33)^2 = 0.109\).

The coefficient of variation of service time;

\[ \frac{6}{22} = 0.27 \quad (9.5) \]

so \(\nu_s = (c.v)^2\) of service time = \((0.27)^2 = 0.074\)

From the tables of waiting times (table III in Appendix 27) the average waiting time in units of average service time (\(w\)) can be read. Linear interpolation on \(\nu_a\) gives \(w = 0.0996\). The average waiting time per mega vessel would be \(22*0.0996 = 2.19\) hours.

The occupancy rate of the Mega vessel berth during these two situations is 61% and rather high. Figure 9.2 shows the average waiting time of a vessel, in units of average service time, in the queue in relation to the occupancy rate of the berth.
Figure 9.2, The relationship between average ship waiting time and the berth occupancy rate

9.3.2 Feeder berths

Concerning the feeder berths a few situations using different arrival time and service time distributions are looked at and approximations of waiting times are given.

**M/D/2**
- Negative exponential inter arrival time distribution
- Deterministic service time distribution
- 2 Berths

**E_r/E/2**
- Erlang distribution for the inter arrival times
- Erlang distribution for the service times
- 2 Berths (n=2)

**M/D/2**
The negative inter arrival distribution (N.E.D.) is used to model inter arrival times when arrivals are completely random.
The service times are assumed to have no variability, that is $\sigma^2 = 0$, which means all the service times are assumed the constant value $1/\mu$.
It is assumed that the service time of the feeder is 12 hours including the mooring, and on the average 1 feeder arrives in 12 hours. Values of waiting times can be obtained from tables.

The utilisation is;

$$u = \frac{\rho}{n} = 50\%$$  \hspace{1cm} (9.6)

From table I can be obtained that the $w=0.176$. The average waiting time will be $w *$ average service time = 2.12 hours.

The average turn around time of a feeder will be 14.12 hours.
**Ek/El/2**

In reality the service times of the feeders will not be deterministic. Especially with variation in the number of containers per feeder the service times will differ. The average service time is assumed to be 12 hours, including the mooring with a standard deviation of 5 hours. In this scenario the average inter arrival time of the feeders is 12 hours with a standard deviation of 7 hours.

An approximation of the value of average waiting time is given based on the linear interpolation on \( \nu_s \) and \( \nu_s \), using the queuing systems: M/M/2, D/M/2 and M/D/2. Let \( W_2(\nu_s, \nu_s, u) \) be the average waiting time in E\(_k\)/E\(_l\)/2 with utilisation \( \lambda_i \), \( \lambda_i^* = \nu_s \) and \( \lambda_i^* = \nu_s \), then

\[
W_2(1,1,u) = \text{the average waiting time in M/M/2 with utilisation } u;
\]
\[
W_2(0,1,u) = \text{the average waiting time in D/M/2 with utilisation } u;
\]
\[
W_2(1,0,u) = \text{the average waiting time in M/D/2 with utilisation } u;
\]

The average waiting time in general case is given by:

\[
W_2(\nu_s, \nu_s, u) = (1- \nu_s) \nu_s^* W_2(0,1,u) + \nu_s (1- \nu_s)^* W_2(1,0,u) + \nu_s \nu_s^* W_2(1,1,u) \tag{9.7}
\]

The average waiting time in M/M/2 can be calculated using formula 9.8.

\[
W = \frac{1}{n \mu} \frac{\rho^n}{n!} \frac{P(0)}{(1-\rho/n)^2} \tag{9.8}
\]

where

\[
P(0) = \left[ 1 + \rho + \frac{\rho^2}{2!} + \frac{\rho^3}{3!} + \cdots + \frac{\rho^n}{n!(1-\rho/n)} \right]^{-1} \tag{9.9}
\]

Using (9.8) and (9.9) the average waiting time \( W_2(1,1,u) = 0.333 \times \text{average service time} \).

The average waiting time in D/M/2 can be obtained from table II, \( W_2(0,1,u) = 0.065 \times \text{average service time} \).

The average service time in M/D/2 obtained from table I = 0.176 \times \text{average service time}.

With:
- \( \nu_s = (7/12)^2 = 0.34 \)
- \( \nu_s = (5/12)^2 = 0.17 \)

in formula (9.7) results in:

\[
W_2(\nu_s, \nu_s, u) = (1 - 0.34) \times 0.17 \times 0.0649 + 0.34 \times (1 - 0.17) \times 0.176 + 0.34 \times 0.17 \times 0.333 = 0.076
\]

0.076 \times \text{average service time} = 0.91 \text{ hours average waiting time per feeder.}

### 9.3.3 Yearly throughput

Using the results of the queuing system, something can be said very roughly about a yearly throughput at the terminal. The Assumptions made are shown below;

**Assumptions:**
- The mooring time of the vessels is included in the service time
- Erlang distribution for the inter arrival times mega vessels (48;12)
- Erlang distribution for the service times of the of the mega vessels (22;6)
- Erlang distribution for the inter arrival times feeders (12;7)
• Erlang distribution for the service times of the of the mega vessels (12, 5)
• 1 Berth available for the mega vessels
• 2 berths available for the feeder vessels
• production per portainer (moves per hour) 55.7 unloading and 50 loading
• Average number of moves per mega vessel (6300 loading, 3150; unloading, 3150, loading)
• 6 portainers per mega vessel
• 360 days per year 24 hours per day

Demand;
The service level = average waiting time/average service time: 0.1

Result;
Using these assumptions with the queuing theory the yearly throughput can be calculated;

360 (days)*24 (hours) = 8640 hours per year is the terminal operational. On the average once in the 36 hours a mega vessel arrives at the terminal. In total 240 mega vessels arrive during the year.
During each mega vessel call 6300 container moves are made. In total this is 6300*1.5=9450 mega TEU per mega vessel call.
In a year 240*9450=2,268,000 TEU is transhipped using the mega vessel berth.
On the other two berths the same amount is transhipped, the total amount would be 4,536,000 TEUs for the terminal or an average of 1,512,000 TEUs per berth.

The mean occupancy rate for the mega vessel berth is 61%, the utilisation of the feeder berths is 50%.
The service level for the mega vessels is:

\[
\frac{2.19}{22} = 0.099
\]

The service level for the feeder is:

\[
\frac{0.91}{12} = 0.075
\]

These service levels are within the boundary condition.

Compared to the most advanced existing transhipment terminal these values are high. For example:
Every berth at the Pasir Panjang Terminal in Singapore handled an average of 750,000 TEUs in the year 1998.
10 Conclusions

The reference container terminal as proposed in this study can be realised from a logistic point of view.
- The horizontal transportation system has not shown unacceptable congestion rates during any of the simulation runs based on the reference terminal.
- The container storage system has shown no unacceptable congestions of rgy's waiting on the warehouse lanes.
- The length of rgy row's under the portainers is acceptable while maintaining a high occupancy rate for the portainers.

Conclusions for each component on the terminal are as follows:

Handling strategies
The optimal handling strategy for the reference terminal is unloading a container vessel in the yard located behind its berth and loading a container vessel from the yard located behind the adjacent berth.

The shuffling of containers is a time consuming process while the advantages are small. The difference between the adjacent loading and the fast loading of a container vessel are marginal.

Berth productions on Berth 1 & 3
The undisturbed berth productions on the reference terminal for the feeder vessel berths vary from 236 moves per hour for unloading to 206 moves/hour for loading from the adjacent berth, Yard 2. This production rate results in a service time of 9.6 hours for the standard feeder vessel used in this study.

The berth productions on the reference terminal for the feeder vessel berths during maximum occupation on the terminal vary from 231 moves/hour for unloading to 194 moves/hour for loading from the adjacent berth, Yard 2. This results in a maximum service time of 9.9 hours for the standard feeder vessel.

The handling rates on the feeder vessel berths have been achieved with 4 portainers per berth.

Berth productions on Berth 2
The undisturbed berth productions on the reference terminal for the mega vessel berth vary from 352 moves per hour for unloading to 324 moves/hour for loading from the adjacent yards, Yards 1 or 3. This production rate results in a service time of 18.6 hours for the standard mega vessel used in this study.

The berth productions on the reference terminal for the mega vessel berths during maximum occupation on the terminal vary from 344 moves/hour for unloading to 300 moves/hour for loading from the adjacent berth, Yards 1 or 3. This results in a maximum service time of 19.8 hours for the standard mega vessel.

The handling rates on the mega vessel berth have been achieved with 6 portainers.

Yearly throughput
Compared to the most advanced existing transhipment terminal the calculated yearly throughput of 1,512,000 TEU per berth is very high. For example:
Every berth at the Pasir Panjang Terminal in Singapore handled an average of 750,000 TEUs in the year 1998.
Number of rgv's
6 rgv's per portainer during the unloading of a container vessel is sufficient, as the portainer occupancy rates remain high. During the loading of a container vessel from an adjacent yard the number of 10 rgv's is low. 12 rgv's per portainer will provide higher portainer occupancy rates thereby increasing the handling rates.

Warehouse cranes
The reference terminal is equipped with 70 warehouse cranes. An occupancy rate calculation has been made based on the output of the computer simulations. By applying a 70 % occupancy rate to the warehouse cranes during the normative configuration, this results to 16 warehouse cranes in the 20’ stacks and 18 warehouse cranes in the 40’ stacks.

Applying variable handling cycle times with a mean cycle of 80 seconds instead of a deterministic handling cycle of 80 seconds decreases the terminal productions with about 4 %.

Land usage
The land usage of the reference terminal as simulated and without space reserved for the rgv waiting rows is 175.5 m * 884 m = 15.5 ha.

The land usage of the terminal included the rgv waiting capacities is 185 m * 884 m = 16.5 ha.
This is the area without rgv parking and service buildings.
11 Recommendations

Annual berth production
In order to determine the annual berth production of the terminal longer simulation runs should be initiated. This model is too detailed to simulate these longer periods. The level of detail should be reduced before periods as long as months can be simulated.

- The actual driving of the rgv's on their lanes should not be included in the model as this asks too much of the memory space of the computer simulation program. The driving could be replaced by time penalties. The duration of these time penalties depends on the situation on the terminal at that moment. Their values can be gathered from runs of the current model.
- The terminal must be linked to a vessel arrival scheme to determine the turn-around times of the vessels and to determine the needed stack capacities.
- A simulation run with less than the maximum number of 14 portainers can be carried out carrying out the portainer assigning process described in Chapter 6.2.5.

PROSIM should be able to be supported by animation, which would make it more apprehensible for outsiders. The effects of different parameter changes are than easily spotted. For the current model animation is not an option as the program runs too slow.

Based on these longer runs an analysis can be made to determine the needed number of portainers and the needed stack capacity.

Cost calculation of the project
This study has not been concentrated on the costs. Obviously the costs of the project are very important. A cost analysis for this project including the civil structures needed for the terminal components like the foundation should be made in order to determine the feasibility of the terminal.

Properties of the rgv
In this study the acceleration of the rgv's is assumed to be 1 m/s². This is an assumption and research should prove whether this is feasible when using linear motor based transfer technology.

The disc rotation used to turn the wheels of the rgv when it has to make a 90° turn takes 4 seconds. This is an assumption, research should conclude whether this is possible.

Rgv-parking
Not all available rgv's will be active constantly. For this reason rgv parking is needed. The positioning should be determined and added to the layout of the terminal.

Horizontal transportation layout
The reference terminal layout consists of 2 pairs of inter-berth lanes. A simulation run with 1 pair of inter-berth lanes should determine if 2 pairs are necessary.

Container Warehouses
The maximum stacking height has, quite arbitrary, been assumed to be 12 containers high. A calculation for finding the economic optimum between stacking height and number of warehouses should be made. An important parameter is to what degree the land area is
scarce. If, as in Singapore, massive land reclaiming projects are needed, increasing stacking height will be cost effective.

**Service points**
The reference terminal contains 3 service points per stack-lane. These service points all have to host a shuttle in order to retrieve the containers from the visiting rgv's. A study should be made in order to find the optimum number of service points per warehouse lane.

**Shuttles**
In the terminal shuttles are used to transport a container between the warehouse crane, a service point and the rgv. The shuttle is assumed to transport a container in 15 seconds from a service point onto a warehouse crane, research should conclude whether this is possible.

**Warehouse cranes**
The reference terminal consists of 70 warehouse cranes. The occupation rate of these cranes is low. Calculations suggest the terminal could function well with less than 70 cranes if the cranes interchange more between the warehouses. An assumption has been made that an occupancy rate of 70% is acceptable. Simulation runs should be carried out to find the maximum acceptable warehouse crane occupancy rate.

**Portainer cycle times**
This study has been carried out using portainer cycle times based on the FAMAS study. Different portainer cycle times of different portainer systems could be implemented in the program to check the reactions of the terminal.

**Optimistic results**
The simulation results are very optimistic as a result of only small disturbances in the terminal process. The process runs very smooth.

The warehouse crane cycle times in the experiments used are deterministic cycle times. The experiment with variable crane cycle times proved that a variable crane cycle time with the same mean does not produce the same handling rates on the terminal. Different warehouse crane cycle times in combination with disturbances in the portainer cycle times and disturbances in the horizontal transport should be implemented in the program.

**Twin lift**
If a twin lift system is applied for the 20 feet containers the portainer production [TEU/move] could improve from 1.5 TEU/move to 2 TEU/move. The number of rgv's that transports 20' containers between the stacks and the berths will decrease. The warehouse crane occupancy rates will increase as a result.

**Logistic control**
As the terminal is almost completely automated, the control of the full range of terminal activities is a key condition for the realisation of this terminal. This is a very complicated logistic issue, which should be investigated for feasibility.
12 References


[L.4] Drewry, Post-panamax containerships 6000 TEU and beyond, september 1996


[L.6] Ligteringen H.; Container handling in Seaports; Pianc conference, September 1997


[L.12] Bakelaar, A.D.; Crane feeding system; Vakgroep Transporttechnologie, report 98.3.TT.5044; TU Delft June 1998


[L.14] Carder, F.B., a.o; Apparatus for marine cargo container handling and storage, U.S. patent number: 5.540.532, July 1996.

Internet site:

http://www.psa.com.sg
Study on the Design and Simulation of a High capacity transshipment terminal

- Appendices 1-27 & 29

Impression of the portainers and container warehouse

December 2000
Hein van Hees
Thies Visscher

TU Delft
Delft University of Technology

Faculty of Civil Engineering and Geosciences
Department of Hydraulic and Geotechnical Engineering
Section of Hydraulic Engineering
Appendices

Appendix 1: Lay out of the reference terminal ......................................................... 3
Appendix 2: Node numbers and stacknumbers of the reference terminal .............. 7
Appendix 3: Annual berth production based on vessel arrival pattern .................... 11
Appendix 4: Container storage capacity calculation ............................................. 13
Appendix 5: Number of warehouse rows and stacking height ................................. 15
Appendix 6: Lane capacity calculations ............................................................... 17
Appendix 7: The rgv flows at the terminal ............................................................ 19
Appendix 8: Warehouse crane capacity calculation ............................................. 27
Appendix 9: Warehouse crane occupancy rates .................................................... 29
Appendix 10: Determining the number of rgv's at the terminal ......................... 33
Appendix 11: Flow chart of the simulation program ............................................. 37
Appendix 12: The step by step trace of a rgv ....................................................... 41
Appendix 13: Input parameters for the experiment with one feeder vessel at Berth 1, unloading & Fast Loading ................................................................. 43
Appendix 14: Input parameters for the experiment with a feeder at Berth 1, loading containers from an adjacent berth ................................................................. 49
Appendix 15: Input parameters for the experiment "shuffling from Berth 1" ........... 55
Appendix 16: Input parameters for the experiment with one mega vessel at Berth 2 unloading ................................................................. 59
Appendix 17: Input parameters for the experiment with one mega vessel at Berth 2, loading ................................................................. 65
Appendix 18: Input parameters for the experiment with one mega vessel at Berth 2 unloading and loading containers from an adjacent berth ................................. 67
Appendix 19: Input parameters for the experiment "shuffling from Berth 2" ............ 75
Appendix 20: Input parameters for the experiment adjacent loading at all three berths 83
Appendix 21: Input parameters for the experiment adjacent loading at Berth 1 and 3 and unloading at Berth 2 ................................................................. 103
Appendix 22: Input parameters for the experiment adjacent loading on Berth 2, unloading on Berths 1 & 3 ................................................................. 115
Appendix 23: Input parameters for the experiment "shuffling from 3 berths" .......... 125
Appendix 24: Input parameters for the experiment adjacent loading at Berth 1 and 3 using 12 rgv's and unloading at Berth 2 ................................................................. 133
Appendix 25: Input parameters for the experiment with disturbances in the portainer cycle times. ................................................................. 145
Appendix 26: Input parameters for the experiment with variable crane cycle times.. 151
Appendix 27: Table section ................................................................................... 155
Appendix 28: Listing computer program .............................................................
Appendix 29: Layout including rgv waiting spaces ............................................. 160
Appendix 1: Lay out of the reference terminal
Appendix 2: Node numbers and stacknumbers of the reference terminal
Appendix 3: Annual berth production based on vessel arrival pattern

- **Calculation for mega vessels moored at Berth 2:**

  Average portainer production: 60 moves/hour  
  Number of portainers on Berth 2: 6 portainers  
  TEU factor: 1.5  
  Operational days in a year: 365 days  
  Amount of TEU on a mega vessel: 8000 TEU  
  Desired berth occupancy rate: 50 %  
  Terminal share for unloading & loading: 60 %  

  Total amount of TEU per call: \(8000 \times 0.6 \times 2 = 9450\) TEU

  Handling hours: \(\frac{9450}{60 \times 6 \times 1.5} = 17.5\) hours

  Including maneuvering and mooring: 19.5 hours

  Arrival scheme: One mega vessel every \(\frac{19.5 \times 365}{(365 \times 24) \times 0.5} = 1.6\) days

  Assume an arrival frequency of 1 mega vessel every 1.5 days

  The berth occupancy rate will then amount to: \(\frac{365 \times 19.5}{1.5 \times 365} = 54\%\)

  Total of handled TEU on Berth 2: \(\frac{60 \times 6 \times 1.5 \times 365 \times 17.5}{1.5} = 2.3\) million TEU's/year

- **Calculation for feeder vessels moored at Berth 1 & 3**

  Average portainer production: 60 moves/hour  
  Number of portainers on Berth 2: 4 portainers  
  TEU factor: 1.5  
  Days in a year: 365 days  
  Amount of TEU on a feeder vessel: 1575 TEU  
  Desired berth occupancy rate: 50 %  
  Terminal share for unloading & loading: 100 %  
  Total amount of TEU per call: 3150 TEU

  Handling hours: \(\frac{3150}{60 \times 4 \times 1.5} = 8.75\) hours

  Including maneuvering and mooring: 10.75 hours

  Arrival scheme per berth: One feeder vessel every \(\frac{10.75 \times 365}{(365 \times 24) \times 0.5} = 0.9\) days
Appendix 3

Assume an arrival frequency of 1 feeder vessel every day for each berth

The berth occupancy rate will then amount to:

\[
\frac{\left(\frac{365 \times 10.75}{0.5}\right)}{365 \times 24 \times 2} = 44.8\%
\]

Total of handled TEU on Berth 1&3:

\[
\frac{2 \times 60 \times 4 \times 1.5 \times 365 \times 8.75}{1} = 2.3 \text{ million TEU's/year}
\]

The total annual amount of handled TEU on Berth 2 is the same as the total annual amount of handled TEU on Berth 1 & 3. This of course is no coincidence. By carrying out an iteration process with the arrival frequency of the feeder vessels and their carrying capacities as variables the annual throughputs on Berth 2 and Berths 1 & 3 can be kept equal. This has resulted in a feeder vessel capacities of 1575 TEU.
Appendix 4: Container storage capacity calculation

The calculation of the storage capacity of the stack has been carried out based on the method as described in Chapter 3.7:

The used formula:

\[ C_o = q_u + \frac{q_u(D_u - p)}{D_u} + \frac{q_u(D_u - 2p)}{D_u} + \ldots + \frac{q_u(D_u - np)}{D_u} + q_i + \frac{q_i(D_i - (p-1))}{D_i} + \frac{q_i(D_i - (2p-1))}{D_i} + \ldots + \frac{q_i(D_i - (np-1))}{D_i} \]

With:
- \( q_u \) = number of TEU to be unloaded from a mega vessel
- \( q_i \) = number of TEU loaded onto a mega vessel
- \( D_u \) = number of days that the feeder vessels are allowed to deliver containers in advance for the mega vessel
- \( D_i \) = number of days that the feeder vessels are allowed to retrieve containers after the arrival of the mega vessel
- \( p \) = periodic interval of mega vessels in days
- \( n = 1, 2, 3, \ldots \)

Resulting in:

\[ C_o = 4800 + 4800 \left( \frac{5 - 1.5}{5} \right) + 4800 \left( \frac{5 - 3}{5} \right) + 4800 \left( \frac{5 - 4.5}{5} \right) + 4800 \left( \frac{5 - 1}{5} \right) + 4800 \left( \frac{5 - (3-1)}{5} \right) + 4800 \left( \frac{5 - (4.5-1)}{5} \right) + 4800 \left( \frac{5 - (6-1)}{5} \right) = 24,000 \text{ TEU} \]
Appendix 5: Number of warehouse rows and stacking height

The number of warehouses and the stacking height is directly related to each other. Increasing the number of warehouses will subsequently decrease the stacking height, but will increase the number of required warehouse cranes, service points and terminal area. Between these relations an economic optimum should be calculated. In this study the terminal area will be normative.

The required stack capacity has been calculated to amount to 24,000 TEU in Chapter 3. The maximum stacking height has, quite arbitrary, been assumed to be able to amount to 11 containers or 12 stories in height. The following formula has been used.

\[ S = \frac{C}{W \times B \times L} \]

- \( S \) = stacking height
- \( C \) = required stack capacity (= 24,000 TEU)
- \( W \) = number of warehouse units (= 70)
- \( B \) = number of TEU distributed over width of the warehouse (= 4)
- \( L \) = number of TEU distributed over the length of the warehouse (= 8)

6 warehouse rows
Using 6 warehouse rows the total amount of warehouse units is: 84. This will amount to a stacking height of 8.9 containers. Including a small shuffling buffer, as containers are stored two units’ deep storage height amounts to 9-10 containers.

5 warehouse rows
Using 5 warehouse rows the total amount of warehouse units is: 70. This will amount to a stacking height of 10.71 containers. Including a small shuffling buffer, as containers are stored two units’ deep storage height amounts to 11-12 containers.

In this study the choice has been made to simulate the terminal with 5 warehouses resulting in a stacking height of 12 containers. As emphasis of this study has not been economic or constructive but a logistic one, a recommendation will be made for a further analysis on stacking height.
Appendix 6: Lane capacity calculations

The lane capacity can be calculated by determining the minimal accepted time between 2 succeeding rgv's driving on the same lane. This follow up time can be established by determining the time a rgv requires for entering a rgv lane.

Yard Lane capacity
The capacity of the yard lane is determined by the time it takes for a stationary rgv to enter the yard lane and accelerate to the required 1 m/s. This process can be seen in Figure 6.1.

![Diagram of rgv entrance onto the Yard lane](image)

*Figure 6.1 rgv entrance onto the Yard lane*

At T=0 seconds the rgv is about to accelerate and decelerate from its position on for instance a warehouse lane onto the yard lane. Eight seconds later on T=8 s. the rgv has just decelerated to 0 m/s and is about to have its wheels turned 90°. After 4 seconds the wheels are turned and the rgv is about to accelerate to 1 m/s. This takes one second.

The yard lane capacity can now be determined by calculating the necessary follow up time between the hearts of 2 succeeding rgv's. The follow up time is: 15.5 seconds. The yard lane capacity thus is almost 4 rgv's per minute.
Inter-berth Lane capacity
The capacity of the Inter-berth lanes is determined in a similar fashion. By calculating the required time for a rgv to enter the inter-berth lane, have its wheels turned and accelerate to 3 m/s the follow up time between the hearts of two rgv’s can be determined as can be seen in Figure 6.2.

![Diagram of rgv entrance onto the inter-berth lane](image)

Figure 6.2, rgv entrance onto the inter-berth lane

Accelerating with 1 m/s\(^2\) onto the inter-berth lane takes the rgv 3,5 seconds. The 90°-wheel rotation takes 4 seconds. Accelerating to 3 m/s will bring the total amount required time to 10,5 seconds. The heart to heart follow up time can be determined as 43 meters. Dividing this distance by the speed on the inter-berth lane comes down to a follow up time of 14,3 seconds, or about 4 rgv’s per minute.
Appendix 7: The rgv flows at the terminal

Calculations concerning the expected rgv flows at the terminal.

After having calculated the capacities of the two different rgv lanes in Appendix 6, it is possible to analyse the different situations at the terminal and to judge whether the terminal can handle the situation in terms of capacity.

In this appendix the terminal is schematised; only the important nodes and lanes are shown and a few situations are considered. The terminal is shown in Figure 7.1. The nodes at the top are the nodes that connect the yard-lanes with the inter-berth lanes located behind the yards. The numbers refer to the node numbers.

![Figure 7.1, the terminal](image)

The green nodes indicate an entrance onto the yard-lanes. At the red nodes the flow of rgv's can leave the yard lane and enter the inter-berth lanes located behind the yard. The individual warehouse lanes are not shown in this figure. The inter-berth lanes are connected to the yard-lanes by shortcuts. The black dots are the portainer locations where the portainers are situated, as can be seen in Figure 7.1, there are two ways to get to one portainer. In this appendix the node numbers are not important but the mean rgv flows are. Therefore the terminal is further schematised as can be seen in Figure 7.2.
The black arrows refer to mean rgv flows.

Using this schematisation, four situations will be evaluated.

- One feeder vessel is moored at Berth 1 and is being unloaded.
- Three vessels are berthed. All of them are being loaded with containers from adjacent berths.
- Three vessels berthed, the two feeder vessels are being loaded with containers from adjacent berths and the mega vessel is being unloaded at Berth 2.
- Three vessels berthed, 1 feeder vessel is being unloaded at Berth 1, the 2 other vessels are being loaded at Berth 2 and 3 with containers from adjacent berths.

**Situation 1: one feeder vessel moored at Berth 1 during unloading.**

The container flows during the unloading of a feeder vessel moored at Berth 1 are restricted to the berth where the vessel is moored. The flows are realised by the assumption that portainers have a mean portainer cycle time of 1 rgv per minute. Knowing that two portainers are connected to a yard-lane means 2 rgv’s per minute will have to pass the node at the end of the portainer locations. This is shown in Figure 7.3.
Appendix 7

This situation should cause no problem. The capacity of the yard lanes is about 4 rgv's per minute.

Situation II: three vessels moored while all are loading from adjacent berths

This situation is more complex than the previous one. First the needed rgv flows will be examined per individual vessel. At the end of this situation description they will be integrated, in order to get an overall picture of the total rgv flows.

Feeder vessel moored at Berth 1

The feeder vessel moored at Berth 1 is being loaded by rgv's receiving their containers from Yard 2. The rgv's will drive to Yard 2 using the two inter-berth lanes. In this case they will use the top inter-berth lanes and the bottom inter-berth lanes. With a mean portainer production of one move per minute this means the rgv flow on the top inter-berth lanes is 2 rgv moves per minute in both directions and the rgv flow on the bottom inter-berth lanes is 2 rgv moves per minute in both directions. The rgv's enter these lanes by using the red nodes. At the top of the yard two yard-lanes exit and connect to the inter-berth lanes. One rgv exits the yard-lane and enters the inter-berth lanes per minute. These are the red nodes (1/2*2).

The yard has three yard-lane exit possibilities for the rgv’s to enter the bottom inter-berth lanes. The mean rgv flows per minute are shown in Figure 7.4.

Figure 7.1, the mean flows at Berth 1 created by rgv's during adjacent loading at Berth 1
Mega vessel moored at Berth 2.

The mega vessel moored at Berth 2 is being loaded with containers retrieved from the adjacent berths, Berth 1 and 3. The flows look like the ones created by the rgv’s working for the feeder berths. The flows are shown in Figure 7.7.

![Diagram of container flows](image)

*Figure 7.7 mean flows created at Berth 2 by rgv’s loading at Berth 2*

The rgv’s from Berth 2 retrieve their containers from Berth 1 and 3. This is shown in Figure 7.8.

![Diagram of container flows](image)

*Figure 7.8, flows created by rgv’s from Berth 2 collecting containers behind Berth 1 & 3*
Appendix 7

Integrating all the r gv flows discussed above gives a total r gv flow pattern at the terminal of the 3 vessels loading from adjacent berths. This total flow is shown in Figure 7.9.

Figure 7.9, mean total flow of r g v's when 3 vessels are being loaded from adjacent berths

As the yard lanes and the inter-berth lanes have a lane capacity of about 4 r g v's per minute the designed layout should work.
Situation III: three vessels moored with two vessels loading from an adjacent berth and one unloading

In this case a mega vessel is being unloaded by its portainers by delivering the unloaded containers to the stacks in Yard 2. Feeder vessels moored at Beth 1 and Berth 3, are being adjacent loaded from the same yard: Yard 2. This combination will put great pressure on the yard-lanes of Yard 2. When a vessel is being handled, the portainers handling the vessel have a mean cycle of one container per minute. The total of container flows can be seen in Figure 7.10.

![Diagram of container flows](image)

*Figure 7.10, total flow when Berth 2 is being unloading and Berth 1 & 3 are being loaded from the adjacent Yard 2.*

The nodes connecting the portainers to the yard-lanes are very busy and demand a capacity of 10/3 rgy per minute. The capacity of these lanes is about 4 rgy's per minute. The computer simulation will have to point out whether or not the terminal is capable of handling this many rgy's a minute.

In this situation there is the possibility to make more use of the top inter-berth lanes and less use of the bottom inter-berth lanes. This will relieve the pressure on the nodes behind the portainers. A balance could be found.
Situation IV: three vessels moored, with Beth 1 unloading and Berth 2 & 3 adjacent loading.

In this situation, three vessels are moored at the berths. The feeder vessel at Berth 1 is being unloaded and the mega vessel at Berth 2 and the feeder vessel at Berth 3 are being loaded with containers from the adjacent yards. The total flow of rgv's can be seen in Figure 7.11.

Figure 7.11, three vessels at the berth; unloading at Berth 1 adjacent loading at Berth 2 and 3.
Appendix 8: Warehouse crane capacity calculation

This calculation is made in order to get a first impression of the production capacity of the warehouse crane. Some assumptions have been made in order to simplify this calculation.

Assumptions

- Horizontal movements will take place during vertical transportation, implying no extra time will be taken into account.
- Shuffling time will be left out of the calculation. If the terminal operator has in advance information on vessel schedules and vessel unloading and loading schemes, he can anticipate on his actions and minimise the amount of shuffling moves.
- The stacking height in the warehouse will be 12 containers high, making it a 13-storey warehouse when including the ground floor where the warehouse lanes and the service points are placed.
- The operating speeds of the warehouse cranes are assumed to have the following properties.
  
  - Crane travelling speed: 2.5 m/s
  - Lift speed:
    - Full up: 0.5 m/s
    - Empty up/down: 0.83 m/s
    - Full down: 0.83 m/s
  - Shuttle speed: 0.5 m/s
  - Acceleration: 0.5 m/s²

Calculation

Maximum lifting height = Storey height * Number of stories = 3 * 11 = 33 m.

- Lifting up (full):
  
  Acceleration time: \( t_a = \frac{v}{a} = \frac{0.5}{0.5} = 1 \) second.
  
  Acceleration distance: \( X_a = \frac{1}{2}a t^2 = 0.25 \text{ m} \Rightarrow 2 * 0.25 = 0.5 \text{ m} \).
  
  Full speed lifting: \( 33 \text{ m} - 0.5 \text{ m} = 32.5 \text{ m} \Rightarrow t_f = 32.5/0.5 = 65 \text{ seconds} \).
  
  Total lifting time: \( T_l = 65 + 2 = 67 \text{ seconds} \).

- Lifting up (empty):
  
  Acceleration time: \( t_a = \frac{0.83}{0.5} = 1.66 \) seconds.
  
  Acceleration distance: \( X_a = \frac{1}{2}a t^2 = 0.69 \text{ m} \Rightarrow 2 * 0.69 = 1.38 \text{ m} \).
  
  Full speed lifting: \( 33 \text{ m} - 1.38 \text{ m} = 31.62 \text{ m} \Rightarrow t_f = 31.62/0.83 = 38.1 \text{ seconds} \).
  
  Total lifting time: \( T_l = 38.1 + 3.3 = 41.4 \text{ seconds} \).

- Lowering takes the same amount of time as lifting up empty \( \Rightarrow 41.4 \text{ seconds} \).

Shuttle:

- 1 rack deep (2.6 m).
  
  Acceleration time: \( t_a = \frac{0.5}{0.25} = 2 \) seconds.
  
  Acceleration distance: \( X_a = \frac{1}{2}a t^2 = 1 \text{ m} \Rightarrow 2 * 1 = 2 \text{ m} \).
  
  Full speed lifting: \( 2.6 \text{ m} - 2 \text{ m} = 0.6 \text{ m} \Rightarrow t_f = 0.6/0.5 = 1.2 \text{ s} \).
  
  Total operating time: \( T_l = 1.2 + 2 + 2^2 \text{ (jacking up & down)} = 5.2 \text{ seconds} \).

- 2 racks deep (5.8 m).
  
  Acceleration time: \( t_a = \frac{0.5}{0.25} = 2 \) seconds.
  
  Acceleration distance: \( X_a = \frac{1}{2}a t^2 = 1 \text{ m} \Rightarrow 2 * 1 = 2 \text{ m} \).
  
  Full speed lifting: \( 5.8 \text{ m} - 2 \text{ m} = 3.8 \text{ m} \Rightarrow t_f = 3.8/0.5 = 7.6 \text{ s} \).
  
  Total operating time: \( T_l = 7.6 + 2 + 2^2 \text{ (jacking up & down)} = 13.6 \text{ seconds} \).
Situation 1: Retrieving a container out of a rack:
Actions:  
1. Lifting up to a rack.
2. Loading container onto the crane with the shuttle.
3. Lowering the crane.
4. Unloading the container onto the service point.

2 Deep:

<table>
<thead>
<tr>
<th>12th storey (max. height)</th>
<th>6th storey (halfway)</th>
<th>1st storey (lowest)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>41.4</td>
<td>23.3</td>
</tr>
<tr>
<td>2.</td>
<td>23.2</td>
<td>23.2</td>
</tr>
<tr>
<td>3.</td>
<td>41.4</td>
<td>23.3</td>
</tr>
<tr>
<td>4.</td>
<td>10.4</td>
<td>10.4</td>
</tr>
<tr>
<td>Total:</td>
<td>116.4 s</td>
<td>80.2 s</td>
</tr>
</tbody>
</table>

1 Deep:

| Total: | 103.6 s | 67.4 s | 31.4 s |

Average Time for situation 1 = \( \frac{73.9}{s} \)

Situation 2: Depositing a container in a rack:
Actions:  
1. Loading container from a service point onto the crane with the shuttle.
2. Lifting up to a rack.
3. Unloading the container into the rack.
4. Lowering the crane.

2 Deep:

<table>
<thead>
<tr>
<th>12th storey (max. height)</th>
<th>6th storey (halfway)</th>
<th>1st storey (lowest)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>10.4</td>
<td>10.4</td>
</tr>
<tr>
<td>2.</td>
<td>67</td>
<td>37</td>
</tr>
<tr>
<td>3.</td>
<td>23.2</td>
<td>23.2</td>
</tr>
<tr>
<td>4.</td>
<td>41.4</td>
<td>23.3</td>
</tr>
<tr>
<td>Total:</td>
<td>142 s</td>
<td>93.9 s</td>
</tr>
</tbody>
</table>

1 Deep:

| Total: | 131.2 s | 81.1 s | 33.1 s |

Average Time for situation 2 = \( \frac{87.9}{s} \)

Total average time without shuffling: \( \frac{77.1 + 89.9}{2} = \frac{167}{2} = \frac{83.9}{s} \)

Evaluation

The calculated maximum capacity of the yard-lane can amount to 240 moves/hour. Assuming the rgv flow is evenly spread out over each warehouse-lane implicates that each lane must be able to cope with \( 240 \div 10 = 24 \) moves an hour, or one move every 2.5 minutes or one move every 150 seconds.

Subtracting the two averages gives a buffer of \( 150.0 - 80.9 = 69.1 \) seconds. It seems one warehouse crane per warehouse-lane should be sufficient, bearing in mind that these calculations assume that the maximum rgv flow through the yard-lane is evenly spread out over all 10 connecting warehouse lanes. The computer simulation must give a clarification on this assumption.
Appendix 9: Warehouse crane occupancy rates

Unloading/fast-loading of a container vessel.

Based on the assumed 1.5 TEU ratio, 50% of the handled containers will be stored in the 20’ container stack while the remaining 50% will be stored in the 40’ container stack. Separate warehouse crane capacity calculations have been made for the 20’ and 40’ container stack. As the portainer production has been determined to achieve 1 container per minute, both 20’ and 40’ container stacks will receive an average of one container per 30 seconds.

![Diagram](image)

*Figure 9.1, unloading/fast-loading on the terminal*

**Occupancy rate calculation of the 20’ container warehouse crane**

The 20’ container stack located behind an unloading/fast-loading portainer receives a container flow of 30 containers per hour.
A stack has the capacity of one move per 80 seconds (Appendix 8) or 45 moves per hour.
The 20’ container stack contains 2 warehouse cranes.

The average warehouse crane occupancy rate is: \[
\left( \frac{30}{45 \times 2} \right) \times 100\% = 33.3\%
\]

**Occupancy rate calculation of the 40’ container warehouse crane**

The 40’ container stack located behind an unloading/fast-loading portainer also receives a container flow of 30 containers per hour.
A stack has the same capacity of one move per 80 seconds or 45 moves per hour.
The 40’ container stack contains 3 warehouse cranes.

The average warehouse crane occupancy rate is: \[
\left( \frac{30}{45 \times 3} \right) \times 100\% = 22.2\%
\]

**Unloading/fast-loading of a container vessel moored at Berth 1 and adjacent loading of a mega vessel moored at Berth 2.**
Added to the 30 containers an hour that a warehouse crane receives during unloading/fast-loading of a vessel moored at Berth 1, are the empty rgv’s that arrive from Berth 2 in order to retrieve a container at Stack 1. The additional number of rgv’s is based on the portainer production of Berth 2 being 60 moves per portainer per hour or 360 moves from Berth 2 per hour. As the rgv’s retrieve their containers from Berth 1 and Berth 3, 180 moves per hour will be the average number of visiting rgv’s from Berth 2. Divided over the four portainers located on Berth 1, results in 45 rgv’s for both 20’ warehouses and 40’ warehouses.

![Diagram](image)

*Figure 9.2, unloading/fast-loading on Berth 1 and adjacent loading on Berth 2*

**Occupancy rate calculation of the 20’ container warehouse crane**

The 20’ container stack located behind an unloading/fast-loading portainer receives a container flow from Berth 2 of $22.5$ empty rgv’s per hour.

The average warehouse crane occupancy rate is: $33.3\% + \left( \frac{22.5}{45 \times 2} \right) \times 100\% = 58.3\%$

**Occupancy rate calculation of the 40’ container warehouse crane**

The 20’ container stack located behind an unloading/fast-loading portainer receives a container flow from Berth 2 of $22.5$ empty rgv’s per hour.

The average warehouse crane occupancy rate is: $22.2\% + \left( \frac{22.5}{45 \times 3} \right) \times 100\% = 39\%$

**Unloading/fast-loading of a mega container vessel moored at Berth 2 and adjacent loading of two feeder vessels moored at Berth 1 and Berth 3.**

The third situation that may occur on the terminal is when a mega vessel is unloading/fast-loading at Berth 2 and two feeder vessels are adjacent loading on the Berths 1 and 3. The warehouse cranes located in the stack behind Berth 2 will have the highest occupancy rates that can be achieved on this terminal. The basic occupancy rates of the warehouse cranes located in the stack behind Berth 2 is $22.2\%$ for the 20’ stack and $33.3\%$ for the 40’ stack as has been calculated before. Adding to this the adjacent loading rgv’s from Berth 1 and Berth 3 will increase the occupancy rates of the warehouse cranes.
Occupancy rate calculation of the 20' container warehouse crane

The 20' container stack located behind the unloading/fast-loading portainers of Berth 2 receives a flow adjacent loading rgv’s from Berth 1 and 3 of 40 empty rgv’s per hour.

The average warehouse crane occupancy rate is: \[33.3\% + \left(\frac{40}{45 + 2}\right) \times 100\% = 77.8\%\]

Occupancy rate calculation of the 40' container warehouse crane

The 40' container stack located behind the unloading/fast-loading portainers of Berth 2 receives a flow adjacent loading rgv’s from Berth 1 and 3 of 40 empty rgv’s per hour.

The average warehouse crane occupancy rate is: \[22.2\% + \left(\frac{40}{45 + 3}\right) \times 100\% = 51.8\%\]
Appendix 10: Determining the number of rgv’s at the terminal

Figure 10.1 schematisation of the terminal

Figure 10.1 is a schematisation of the terminal with some dimensions. This figure makes it possible to determine the number of rgv’s per portainer needed for the different procedures at the terminal. The procedures are unloading a vessel, loading a vessel and shuffling at the terminal.

Number of rgv’s needed per portainer when loading a vessel from an adjacent berth.

When a feeder vessel is being loaded at Berth 1 with containers retrieved from the yard of Berth 2 there are two routes for the rgv’s to get to the Yard 2. They can take the top inter-berth route or the bottom inter-berth route of the terminal. The mean transportation cycle distance and needed time for the rgv’s to cover this distance are calculated.

The cycle time for a rgv taking the top route from Berth 1 to 2 starting under the portainer at Berth 1 is shown in Table 10.1. In this example the mean distances have been used.

<table>
<thead>
<tr>
<th>Direction</th>
<th>distance [m]</th>
<th>speed [m/s]</th>
<th>time [s]</th>
</tr>
</thead>
<tbody>
<tr>
<td>The rgv has to go to the top inter berth route</td>
<td>110</td>
<td>1</td>
<td>110</td>
</tr>
<tr>
<td>The rgv has to go to the other berth</td>
<td>310</td>
<td>3</td>
<td>103</td>
</tr>
<tr>
<td>The rgv has to go to the stack entrance</td>
<td>60</td>
<td>1</td>
<td>60</td>
</tr>
<tr>
<td>The rgv drives through the stack</td>
<td>60</td>
<td>1</td>
<td>60</td>
</tr>
<tr>
<td>The rgv gets a container</td>
<td></td>
<td></td>
<td>15</td>
</tr>
<tr>
<td>The rgv has to go to the top inter berth route</td>
<td>60</td>
<td>1</td>
<td>60</td>
</tr>
<tr>
<td>The has to go back to its berth</td>
<td>310</td>
<td>3</td>
<td>103</td>
</tr>
<tr>
<td>The goes to portainer to deliver the container</td>
<td>110</td>
<td>1</td>
<td>110</td>
</tr>
<tr>
<td>The rgv is being served by the portainer</td>
<td></td>
<td></td>
<td>15</td>
</tr>
<tr>
<td>Total</td>
<td>1010</td>
<td></td>
<td>636</td>
</tr>
</tbody>
</table>

Table 10.1, the mean rgv cycle time to retrieve a container at Yard 2 for Berth 1, top route.

With a mean portainer cycle time of 60 seconds, the required number of needed rgv’s is 636 [sec] / 60 [sec] or 10.5 rgv’s per portainer.
The cycle time of a rgv taking the bottom route from Berth 1 to 2 and back starting under the portainer at Berth 1 is shown in Table 10.2.

<table>
<thead>
<tr>
<th>Direction</th>
<th>distance [m]</th>
<th>speed [m/s]</th>
<th>time [s]</th>
</tr>
</thead>
<tbody>
<tr>
<td>The rgv goes to the bottom inter berth route</td>
<td>10</td>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>The rgv has to go to the other berth</td>
<td>310</td>
<td>3</td>
<td>103</td>
</tr>
<tr>
<td>The rgv has to go to the stack entrance</td>
<td>60</td>
<td>1</td>
<td>60</td>
</tr>
<tr>
<td>The rgv drives through the stack</td>
<td>60</td>
<td>1</td>
<td>60</td>
</tr>
<tr>
<td>The rgv gets a container</td>
<td></td>
<td></td>
<td>15</td>
</tr>
<tr>
<td>The rgv goes to the bottom inter berth route</td>
<td>60</td>
<td>1</td>
<td>60</td>
</tr>
<tr>
<td>The has to go back to its berth</td>
<td>310</td>
<td>3</td>
<td>103</td>
</tr>
<tr>
<td>The goes to portainer to deliver the container</td>
<td>10</td>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>Total</td>
<td>820</td>
<td></td>
<td>421</td>
</tr>
</tbody>
</table>

Table 10.2, the mean rgv cycle time to retrieve a container at Yard 2 for Berth 1, bottom route.

The number of rgv’s needed would be 421 [sec] / 60 [sec] is 7 rgv’s per portainer.

The number of rgv needed per portainer when unloading a vessel

During the unloading of a vessel a rgv drives from the portainer to a stack in the stack-cluster behind this portainer, drops its container and drives back to the portainer in order to get the next container. These actions and their needed time demands, are shown in Table 10.3. The rgv starts at Berth 1 and is about to leave the portainer.

<table>
<thead>
<tr>
<th>Direction</th>
<th>distance [m]</th>
<th>speed [m/s]</th>
<th>time [s]</th>
</tr>
</thead>
<tbody>
<tr>
<td>The rgv drives to a stack behind the portainer</td>
<td>55</td>
<td>1</td>
<td>55</td>
</tr>
<tr>
<td>The rgv drives through the stack</td>
<td>60</td>
<td>1</td>
<td>60</td>
</tr>
<tr>
<td>The rgv drops a container at a service point</td>
<td></td>
<td></td>
<td>15</td>
</tr>
<tr>
<td>The rgv drives to the portainer</td>
<td>55</td>
<td>1</td>
<td>55</td>
</tr>
<tr>
<td>The rgv gets a container from the portainer</td>
<td></td>
<td></td>
<td>15</td>
</tr>
<tr>
<td>The rgv drives under the portainer</td>
<td>60</td>
<td>1</td>
<td>60</td>
</tr>
<tr>
<td>Total</td>
<td>208</td>
<td></td>
<td>248</td>
</tr>
</tbody>
</table>

Table 10.3, the mean rgv cycle time when unloading a vessel

The number of rgv’s needed would be 248 [sec] / 60 [sec] is 4.1 rgv’s per portainer.
The number of needed rgv's when shuffling

During shuffling on the terminal the rgv's drive between the stacks using inter-berth lanes and yard-lanes. The actions during shuffling and their needed times are shown in Table 10.4. Assumed is a rgv starting behind Berth 1 and about to leave the stack to take the top route to the adjacent yard.

<table>
<thead>
<tr>
<th>Direction</th>
<th>distance [m]</th>
<th>speed [m/s]</th>
<th>time [s]</th>
</tr>
</thead>
<tbody>
<tr>
<td>The rgv drives to the top route</td>
<td>55</td>
<td>1</td>
<td>55</td>
</tr>
<tr>
<td>The rgv has to go to the other berth</td>
<td>310</td>
<td>3</td>
<td>103</td>
</tr>
<tr>
<td>The rgv has to go to a stack entrance</td>
<td>55</td>
<td>1</td>
<td>55</td>
</tr>
<tr>
<td>The rgv drives through the stack</td>
<td>60</td>
<td>1</td>
<td>60</td>
</tr>
<tr>
<td>The rgv drops a container in the stack</td>
<td></td>
<td></td>
<td>15</td>
</tr>
<tr>
<td>The rgv receives a container in the stack</td>
<td></td>
<td></td>
<td>15</td>
</tr>
<tr>
<td>The rgv drives to the top route</td>
<td>55</td>
<td>1</td>
<td>55</td>
</tr>
<tr>
<td>The rgv drives back to its berth</td>
<td>310</td>
<td>3</td>
<td>103</td>
</tr>
<tr>
<td>The rgv drives through the stack</td>
<td>60</td>
<td>1</td>
<td>60</td>
</tr>
<tr>
<td>The rgv drops a container at a service point</td>
<td></td>
<td></td>
<td>15</td>
</tr>
<tr>
<td>The rgv receives a container at a service point</td>
<td></td>
<td></td>
<td>15</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td></td>
<td></td>
<td><strong>551</strong></td>
</tr>
</tbody>
</table>

Table 10.4, The mean rgv cycle time when shuffling.

The cycle time of a rgv is 551 seconds. The crane capacity of the 5 warehouse cranes behind one portainer location is:

- The time needed for one crane to unload and load a rgv is 160 seconds.
- Per minute this results in 60/160 = 0.375 rgv. With a total of 5 cranes behind a portainer location the mean capacity is 5 * 0.375 = 1.875 rgv per minute.
- If an occupancy rate of 50% is desired for the stacking cranes, this would result in 0.5 * 1.875 = 0.9375 rgv per minute.
- With a rgv cycle time of 551 seconds the required number of rgv's is 0.9375 * (551/60) = 8.6 rgv per portainer location when shuffling.
Appendix 12: The step by step trace of a rgv

In this appendix a rgv will be followed during its ride at the terminal. This appendix is realised by stepping through the program using one rgv per portainer while loading a vessel. This rgv starts in stack 62 and is working at berth 2 where a vessel is being unloaded. The entrance node of the rgv is node 292 and the exit node of the rgv is node 302. The rgv is about to leave the stack and is activated in the agvmod from toquay, line 80. In the table below the actions of the rgv, the time and the distances of the different actions are given. Column 3 refers to the module and the line in the module where the rgv gets its assignment. Column 5 in the table refers to the cur_node, an attribute of the rgv. Looking at this column the route of the rgv can be followed.

<table>
<thead>
<tr>
<th>time[s]</th>
<th>rgv action</th>
<th>module/line</th>
<th>ΔT[s]</th>
<th>length[m]</th>
<th>cur_node</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>work exittime of cur_node</td>
<td>rgv/91</td>
<td>8</td>
<td>0</td>
<td>292</td>
</tr>
<tr>
<td>8</td>
<td>work discrotation</td>
<td>rgv/103</td>
<td>4</td>
<td>0</td>
<td>292</td>
</tr>
<tr>
<td>12</td>
<td>work nodedistance3/nodespeed</td>
<td>rgv/104</td>
<td>0</td>
<td>0</td>
<td>292</td>
</tr>
<tr>
<td>12</td>
<td>work nodedistance1/nodespeed</td>
<td>rgv/119</td>
<td>0.5</td>
<td>0.5</td>
<td>292</td>
</tr>
<tr>
<td>12.5</td>
<td>work max(0, (s_length of.....</td>
<td>rgv/121</td>
<td>2</td>
<td>2</td>
<td>292</td>
</tr>
<tr>
<td>14.5</td>
<td>work nodedistance2/nodespeed</td>
<td>rgv/125</td>
<td>0.5</td>
<td>0.5</td>
<td>293</td>
</tr>
<tr>
<td>15</td>
<td>work nodedistance3/nodespeed</td>
<td>rgv/127</td>
<td>0</td>
<td>0</td>
<td>293</td>
</tr>
<tr>
<td>15</td>
<td>work nodedistance1/nodespeed</td>
<td>rgv/119</td>
<td>0.5</td>
<td>0.5</td>
<td>293</td>
</tr>
<tr>
<td>15.5</td>
<td>work max(0, (s_length of.....</td>
<td>rgv/121</td>
<td>11</td>
<td>11</td>
<td>293</td>
</tr>
<tr>
<td>26.5</td>
<td>work nodedistance2/nodespeed</td>
<td>rgv/125</td>
<td>0.5</td>
<td>0.5</td>
<td>294</td>
</tr>
<tr>
<td>27</td>
<td>work nodedistance3/nodespeed</td>
<td>rgv/127</td>
<td>0</td>
<td>0</td>
<td>294</td>
</tr>
<tr>
<td>27</td>
<td>work nodedistance1/nodespeed</td>
<td>rgv/119</td>
<td>0.5</td>
<td>0.5</td>
<td>294</td>
</tr>
<tr>
<td>27.5</td>
<td>work max(0, (s_length of.....</td>
<td>rgv/121</td>
<td>2</td>
<td>2</td>
<td>294</td>
</tr>
<tr>
<td>29.5</td>
<td>work nodedistance2/nodespeed</td>
<td>rgv/125</td>
<td>0.5</td>
<td>0.5</td>
<td>295</td>
</tr>
<tr>
<td>30</td>
<td>work nodedistance3/nodespeed</td>
<td>rgv/127</td>
<td>0</td>
<td>0</td>
<td>295</td>
</tr>
<tr>
<td>30</td>
<td>work nodedistance1/nodespeed</td>
<td>rgv/119</td>
<td>0.5</td>
<td>0.5</td>
<td>295</td>
</tr>
<tr>
<td>30.5</td>
<td>work max(0, (s_length of.....</td>
<td>rgv/121</td>
<td>11</td>
<td>11</td>
<td>295</td>
</tr>
<tr>
<td>41.5</td>
<td>work nodedistance2/nodespeed</td>
<td>rgv/125</td>
<td>0.5</td>
<td>0.5</td>
<td>296</td>
</tr>
<tr>
<td>42</td>
<td>work nodedistance3/nodespeed</td>
<td>rgv/127</td>
<td>0</td>
<td>0</td>
<td>296</td>
</tr>
<tr>
<td>42</td>
<td>work nodedistance1/nodespeed</td>
<td>rgv/119</td>
<td>0.5</td>
<td>0.5</td>
<td>296</td>
</tr>
<tr>
<td>42.5</td>
<td>work max(0, (s_length of.....</td>
<td>rgv/121</td>
<td>2</td>
<td>2</td>
<td>296</td>
</tr>
<tr>
<td>44.5</td>
<td>work nodedistance2/nodespeed</td>
<td>rgv/125</td>
<td>0.5</td>
<td>0.5</td>
<td>297</td>
</tr>
<tr>
<td>45</td>
<td>work nodedistance3/nodespeed</td>
<td>rgv/127</td>
<td>0</td>
<td>0</td>
<td>297</td>
</tr>
<tr>
<td>45</td>
<td>work nodedistance1/nodespeed</td>
<td>rgv/119</td>
<td>0.5</td>
<td>0.5</td>
<td>297</td>
</tr>
<tr>
<td>45.5</td>
<td>work max(0, (s_length of.....</td>
<td>rgv/121</td>
<td>11</td>
<td>11</td>
<td>297</td>
</tr>
<tr>
<td>56.5</td>
<td>work nodedistance2/nodespeed</td>
<td>rgv/125</td>
<td>0.5</td>
<td>0.5</td>
<td>298</td>
</tr>
<tr>
<td>57</td>
<td>work nodedistance3/nodespeed</td>
<td>rgv/127</td>
<td>0</td>
<td>0</td>
<td>298</td>
</tr>
<tr>
<td>57</td>
<td>work nodedistance1/nodespeed</td>
<td>rgv/119</td>
<td>0.5</td>
<td>0.5</td>
<td>298</td>
</tr>
<tr>
<td>57.5</td>
<td>work max(0, (s_length of.....</td>
<td>rgv/121</td>
<td>2</td>
<td>2</td>
<td>298</td>
</tr>
<tr>
<td>59.5</td>
<td>work nodedistance2/nodespeed</td>
<td>rgv/125</td>
<td>0.5</td>
<td>0.5</td>
<td>299</td>
</tr>
<tr>
<td>60</td>
<td>work nodedistance3/nodespeed</td>
<td>rgv/127</td>
<td>0</td>
<td>0</td>
<td>299</td>
</tr>
<tr>
<td>60</td>
<td>work nodedistance1/nodespeed</td>
<td>rgv/119</td>
<td>0.5</td>
<td>0.5</td>
<td>299</td>
</tr>
<tr>
<td>60.5</td>
<td>work max(0, (s_length of.....</td>
<td>rgv/121</td>
<td>11</td>
<td>11</td>
<td>299</td>
</tr>
<tr>
<td>time [s]</td>
<td>rgv action</td>
<td>module/line</td>
<td>ΔT [s]</td>
<td>length [m]</td>
<td>cur_node</td>
</tr>
<tr>
<td>---------</td>
<td>------------</td>
<td>---------------</td>
<td>--------</td>
<td>------------</td>
<td>----------</td>
</tr>
<tr>
<td>96.5</td>
<td>work nodedistance2/nodespeed</td>
<td>rgv/125</td>
<td>0.5</td>
<td>0.5</td>
<td>300</td>
</tr>
<tr>
<td>97</td>
<td>work nodedistance3/nodespeed</td>
<td>rgv/127</td>
<td>0</td>
<td>0</td>
<td>300</td>
</tr>
<tr>
<td>97</td>
<td>work nodedistance1/nodespeed</td>
<td>rgv/119</td>
<td>0.5</td>
<td>0.5</td>
<td>300</td>
</tr>
<tr>
<td>97.5</td>
<td>work max0, (s_length of.....</td>
<td>rgv/121</td>
<td>7</td>
<td>7</td>
<td>300</td>
</tr>
<tr>
<td>104.5</td>
<td>work nodedistance2/nodespeed</td>
<td>rgv/125</td>
<td>0.5</td>
<td>0.5</td>
<td>301</td>
</tr>
<tr>
<td>105</td>
<td>work nodedistance3/nodespeed</td>
<td>rgv/127</td>
<td>0</td>
<td>0</td>
<td>301</td>
</tr>
<tr>
<td>105</td>
<td>work nodedistance1/nodespeed</td>
<td>rgv/119</td>
<td>0.5</td>
<td>0.5</td>
<td>301</td>
</tr>
<tr>
<td>105.5</td>
<td>work max0, (s_length of.....</td>
<td>rgv/121</td>
<td>7</td>
<td>7</td>
<td>301</td>
</tr>
<tr>
<td>112.5</td>
<td>work nodedistance2/nodespeed</td>
<td>rgv/125</td>
<td>0.5</td>
<td>0.5</td>
<td>302</td>
</tr>
<tr>
<td>113</td>
<td>work nodedistance3/nodespeed</td>
<td>rgv/127</td>
<td>0</td>
<td>0</td>
<td>302</td>
</tr>
<tr>
<td>113</td>
<td>work nodedistance1/nodespeed</td>
<td>macro stat./5</td>
<td>0.5</td>
<td>0.5</td>
<td>302</td>
</tr>
<tr>
<td>113.5</td>
<td>work discrotation</td>
<td>macro stat./6</td>
<td>4</td>
<td>0</td>
<td>302</td>
</tr>
<tr>
<td>117.5</td>
<td>work exittime of cur_node</td>
<td>macro stat./16</td>
<td>8</td>
<td>0</td>
<td>302</td>
</tr>
<tr>
<td>125.5</td>
<td>work s_length of cur_section</td>
<td>macro stat./25</td>
<td>19</td>
<td>19</td>
<td>309</td>
</tr>
<tr>
<td>144.5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

In this table the loop the rgv takes in the rgv module becomes clear. This loop is line 109 to line 136 in the rgv module. The rgv gets out of the loop at the moment it reaches its exit node. In this case the exit node is the entrance of the portainer.

After 113 seconds the rgv leaves the yard lane and enters the portainer location. Thereafter it drives to the portainer and after 144 seconds the rgv is under the portainer and ready to receive a container.

The driving of the rgv from node 292 to the portainer is shown in Figure 12.1.

![Diagram of the rgv route](image-url)

*Figure 12.1, the route of the rgv from the stack to the portainer*
Appendix 13: Input parameters for the experiment with one feeder vessel at Berth 1, unloading & Fast Loading

The basic parameters used for the experiment with the feeder vessel moored at Berth 1 during unloading are shown in Table 13.1. There are 1050 containers unloaded from the vessel. The parameters that have changed during the run are shown in Table 13.2. Inplace counter 1 refers to the number of containers in Yard 1 destined to be taken away by a vessel at Berth 1. These containers can be loaded fast.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>fromtob[1,2]</td>
<td>0</td>
</tr>
<tr>
<td>fromtob[3,2]</td>
<td>0</td>
</tr>
<tr>
<td>fromtob[2,1]</td>
<td>0</td>
</tr>
<tr>
<td>fromtob[2,3]</td>
<td>0</td>
</tr>
<tr>
<td>inplacecounter 1</td>
<td>1200</td>
</tr>
<tr>
<td>inplacecounter 2</td>
<td>0</td>
</tr>
<tr>
<td>inplacecounter 3</td>
<td>0</td>
</tr>
<tr>
<td>seed of uniform</td>
<td>67</td>
</tr>
<tr>
<td>seed of arrivaltimes feeders</td>
<td>17</td>
</tr>
<tr>
<td>seed of arrivaltimes big</td>
<td>27</td>
</tr>
<tr>
<td>import generator feeders</td>
<td>1050</td>
</tr>
<tr>
<td>export generator feeders</td>
<td>1050</td>
</tr>
<tr>
<td>import generator big vessels</td>
<td>0</td>
</tr>
<tr>
<td>export generator big vessels</td>
<td>0</td>
</tr>
<tr>
<td>rgv full speed</td>
<td>3</td>
</tr>
<tr>
<td>rgv low speed passing a shortcut</td>
<td>2</td>
</tr>
<tr>
<td>rgv node speed passing a node</td>
<td>1</td>
</tr>
<tr>
<td>rgv acceleration</td>
<td>1</td>
</tr>
<tr>
<td>minimum distance between trolleys</td>
<td>3</td>
</tr>
<tr>
<td>number of warehouse cranes at terminal</td>
<td>70</td>
</tr>
<tr>
<td>number of agv's per portainer used:</td>
<td>6</td>
</tr>
<tr>
<td>- unloading a vessel</td>
<td></td>
</tr>
<tr>
<td>- loading a vessel from different berth</td>
<td>10</td>
</tr>
<tr>
<td>- shuffling at the terminal</td>
<td>10</td>
</tr>
<tr>
<td>stacking crane cycle time</td>
<td>80</td>
</tr>
<tr>
<td>mean portainer cycle time</td>
<td>58</td>
</tr>
</tbody>
</table>

*Table 13.1, Input parameters for experiment: feeder vessel at Berth 1*

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>fromtob[1,2]</td>
<td>1050</td>
</tr>
<tr>
<td>Inplacecounter 1</td>
<td>150</td>
</tr>
</tbody>
</table>

*Table 13.2, output after experiment: feeder vessel at Berth 1 unloading*

In Table 13.2 the parameter fromtob[1,2] indicates 1050 containers are unloaded from the vessel and stacked in Yard 1, these containers are destined for a vessel at Berth 2. Inplace counter 1 = 150 indicates 150 containers are left in Yard 1 destined to be taken away by a vessel from Berth 1. From the 1200 original containers in Yard 1, 1050 are taken away by the vessel.
13.1.1 Warehouse crane capacity calculation: The unloading of a feeder vessel

A feeder vessel moored at Berth 1 will have its containers unloaded in Yard 1. The number of unloaded containers is 1050. Half of the 1050 containers will be delivered to the 40' stacks and the other half to the 20' stacks.

- Required 20' warehouse crane operational time: $1050/2 \times 80$ seconds = 42,000 seconds
- Required 40' warehouse crane operational time: $1050/2 \times 80$ seconds = 42,000 seconds

Number of warehouse cranes in Yard 1:
- Number of warehouse cranes in the 20' stacks in Yard 1 = $4 \times 2 = 8$ warehouse cranes
- Number of warehouse cranes in the 40' stacks in Yard 1 = $4 \times 3 = 12$ warehouse cranes

Needed operational time per warehouse crane:

- $\frac{42,000}{8} = 5,250$ seconds per 20' stack warehouse crane
- $\frac{42,000}{12} = 3,500$ seconds per 40' stack warehouse crane

Simulated unloading time for the feeder vessel: 4.4 hours ⇒ 15,840 seconds

Occupation rates of the warehouse cranes:

20' stack: $\left( \frac{5,250}{15,840} \right) \times 100\% = 33.1\%$

40' stack: $\left( \frac{3,500}{15,840} \right) \times 100\% = 22.1\%$

13.1.2 Warehouse crane capacity calculation: The fast loading of a feeder vessel

A feeder vessel moored at Berth 1 will have its containers fast loaded from Yard 1. The number of to be loaded containers is 1050. Considering that the number of containers that have been retrieved from the 20' stacks and the 40' stacks are the same means that 525 containers will be delivered to both stack types.

- Required 20' warehouse crane operational time: $525 \times 80$ seconds = 42,000 seconds
- Required 40' warehouse crane operational time: $525 \times 80$ seconds = 42,000 seconds

Number of warehouse cranes in Yard 1:
- Number of warehouse cranes in the 20' stacks in Yard 1 = $4 \times 2 = 8$ warehouse cranes
- Number of warehouse cranes in the 40' stacks in Yard 1 = $4 \times 3 = 12$ warehouse cranes

Needed operational time per warehouse crane:

- $\frac{42,000}{8} = 5,250$ seconds per 20' stack warehouse crane
- $\frac{42,000}{12} = 3,500$ seconds per 40' stack warehouse crane
Appendix 13

Required unloading time for the feeder vessel: 4.3 hours ⇒ 15,480 seconds

Occupation rates of the warehouse cranes:

20' stack: \( \frac{5,250}{15,480} \times 100\% = 33.9\% \)

40' stack: \( \frac{3,500}{15,480} \times 100\% = 22.6\% \)
Figure 13.1, rgvqueue at portlocation 1 when unloading and loading a feeder fast

Figure 13.2, rgvqueue at portlocation 2 when unloading and loading a feeder fast

Figure 13.3, rgvqueue at portlocation 3 when unloading and loading a feeder fast

Figure 13.4, rgvqueue at portlocation 4 when unloading and loading a feeder fast
Appendix 13

Figure 13.5, rgv queue on warehouse lane 22

Figure 13.6, rgv queue on warehouse lane 24

Figure 13.7, rgv queue on warehouse lane 26

Figure 13.8, rgv queue on warehouse lane 28
Figure 13.9 RGV queue on warehouse lane 30
Appendix 14: Input parameters for the experiment with a feeder at Berth 1, loading containers from an adjacent berth

The basic parameters used for the experiment with the feeder at Berth 1 when it is loaded with containers from the adjacent berth are shown in Table 14.1. The parameter fromtob[2,1] indicates the number of containers in Yard 2 which can be used to load the feeder at Berth 1. Inplacecounter 1 indicates the number of containers in Yard 1 that can be used to load the feeder at Berth 1. In this experiment the inplacecounter 1 is 0, indicating the vessel at berth 1 will have to be loaded with containers from Yard 2. In this yard more than enough containers destined for berth 2 have been placed. The feeder unloads only 40 containers as can be seen by the import generator. The parameters after the run are shown in Table 14.2.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>fromtob[1,2]</td>
<td>0</td>
</tr>
<tr>
<td>fromtob[3,2]</td>
<td>0</td>
</tr>
<tr>
<td>fromtob[2,1]</td>
<td>1600</td>
</tr>
<tr>
<td>fromtob[2,3]</td>
<td>0</td>
</tr>
<tr>
<td>inplacecounter 1</td>
<td>0</td>
</tr>
<tr>
<td>inplacecounter 2</td>
<td>0</td>
</tr>
<tr>
<td>inplacecounter 3</td>
<td>0</td>
</tr>
<tr>
<td>seed of uniform</td>
<td>67</td>
</tr>
<tr>
<td>seed of arrival times feeders</td>
<td>17</td>
</tr>
<tr>
<td>seed of arrival times big</td>
<td>27</td>
</tr>
<tr>
<td>import generator feeders</td>
<td>40</td>
</tr>
<tr>
<td>export generator feeders</td>
<td>1050</td>
</tr>
<tr>
<td>import generator big vessels</td>
<td>0</td>
</tr>
<tr>
<td>export generator big vessels</td>
<td>0</td>
</tr>
<tr>
<td>rgy full speed</td>
<td>3 m/s</td>
</tr>
<tr>
<td>rgy low speed passing a shortcut</td>
<td>2 m/s</td>
</tr>
<tr>
<td>rgy node speed passing a node</td>
<td>1 m/s</td>
</tr>
<tr>
<td>rgy acceleration</td>
<td>1 m/s²</td>
</tr>
<tr>
<td>minimum distance between trolleys</td>
<td>3 m</td>
</tr>
<tr>
<td>number of stacking cranes at terminal</td>
<td>70</td>
</tr>
<tr>
<td>number of agv's per portainer used:</td>
<td></td>
</tr>
<tr>
<td>- unloading a vessel</td>
<td>6</td>
</tr>
<tr>
<td>- loading a vessel from different berth</td>
<td>10</td>
</tr>
<tr>
<td>- shuffling at the terminal</td>
<td>10</td>
</tr>
<tr>
<td>stacking crane cycle time</td>
<td>80</td>
</tr>
<tr>
<td>mean portainer cycle time</td>
<td>58</td>
</tr>
</tbody>
</table>

Table 14.1: Input parameters for experiment: Feeder at Berth 1 loading from an adjacent berth

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>fromtob[1,2]</td>
<td>40</td>
</tr>
<tr>
<td>fromtob[2,1]</td>
<td>550</td>
</tr>
</tbody>
</table>

Table 14.2, output after experiment: Feeder vessel at Berth 1 loading from an adjacent berth

In Table 14.2, fromtob[1,2] refers to the imported containers. After the run they are located in Yard 1 and destined to be taken away by a vessel from Berth 2. Fromtob[2,1] refers to the containers left in Yard 2 destined to be taken away by a vessel at Berth 1.
14.1.1 Crane occupancy rates

A feeder vessel moored at Berth 1 will have its containers adjacent loaded from Yard 2. The number of to be loaded containers is 1050. Considering that the number of containers that have been retrieved from the 20’ stacks and the 40’ stacks are the same means that 525 containers will be delivered to both stack types.

- Required 20’ warehouse crane operational time: $525 \times 80 \text{ seconds} = 42,000 \text{ seconds}$
- Required 40’ warehouse crane operational time: $525 \times 80 \text{ seconds} = 42,000 \text{ seconds}$

Number of warehouse cranes in Yard 2:
- Number of warehouse cranes in the 20’ stacks in Yard 2 = $6 \times 2 = 12$ warehouse cranes
- Number of warehouse cranes in the 40’ stacks in Yard 2 = $6 \times 3 = 18$ warehouse cranes

Needed operational time per warehouse crane:

- $\frac{42,000}{12} = 3,500 \text{ seconds per 20’ stack warehouse crane}$
- $\frac{42,000}{18} = 2,333 \text{ seconds per 40’ stack warehouse crane}$

Required unloading time for the feeder vessel: 5.1 hours $\Rightarrow 18,360 \text{ seconds}$

Occupation rates of the warehouse cranes:

- 20’ stack: $\left(\frac{3,500}{18,360}\right) \times 100\% = 19.1\%$
- 40’ stack: $\left(\frac{2,333}{18,360}\right) \times 100\% = 12.7\%$
Figure 14.1, portainer 1 queue when loading a feeder from an adjacent Berth

Figure 14.2, portainer 2 queue when loading a feeder from an adjacent Berth

Figure 14.3, portainer 3 queue when loading a feeder from an adjacent Berth

Figure 14.4, portainer 4 queue when loading a feeder from an adjacent Berth
Figure 14.5, stack queue on warehouse lane 61 when adjacent loading a feeder at Berth 1

Figure 14.6, stack queue on warehouse lane 62 when adjacent loading a feeder at Berth 1

Figure 14.7, stack queue on warehouse lane 63 when adjacent loading a feeder at Berth 1

Figure 14.8, stack queue on warehouse lane 64 when adjacent loading a feeder at Berth 1
Figure 14.9, stack queue on warehouse lane 65 when adjacent loading a feeder at Berth 1

Figure 14.10, stack queue on warehouse lane 66 when adjacent loading a feeder at Berth 1

Figure 14.11, stack queue on warehouse lane 67 when adjacent loading a feeder at Berth 1

Figure 14.12, stack queue on warehouse lane 68 when adjacent loading a feeder at Berth 1
Figure 14.13, the rgv queues at node 116 when adjacent loading a feeder at Berth 1

Figure 14.14, the rgv queues at node 114 when adjacent loading a feeder at Berth 1

Figure 14.15, the grv queues at node 290 when adjacent loading a feeder at Berth 1

Figure 14.16, the rgv queues at node 260 when adjacent loading a feeder at Berth 1
Appendix 15: Input parameters for the experiment "shuffling from Berth 1"

The basic parameters used for this experiment are shown in Table 15.1. The shuffle process stops when there are no more containers in Yard 1 to be taken to Yard 2. The vessel has only 40 import containers and only 40 export containers. The parameters after the run are shown in Table 15.2.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>fromtlob[1,2]</td>
<td>1200 containers</td>
</tr>
<tr>
<td>fromtlob[3,2]</td>
<td>0 containers</td>
</tr>
<tr>
<td>fromtlob[2,1]</td>
<td>1600 containers</td>
</tr>
<tr>
<td>fromtlob[2,3]</td>
<td>0 containers</td>
</tr>
<tr>
<td>inplacecounter 1</td>
<td>50 containers</td>
</tr>
<tr>
<td>inplacecounter 2</td>
<td>0 containers</td>
</tr>
<tr>
<td>inplacecounter 3</td>
<td>0 containers</td>
</tr>
<tr>
<td>seed of uniform</td>
<td>67 -</td>
</tr>
<tr>
<td>seed of arrivaltimes feeders</td>
<td>17 -</td>
</tr>
<tr>
<td>seed of arrivaltimes big</td>
<td>27 -</td>
</tr>
<tr>
<td>import generator feeders</td>
<td>40 containers</td>
</tr>
<tr>
<td>export generator feeders</td>
<td>40 containers</td>
</tr>
<tr>
<td>import generator big vessels</td>
<td>0 containers</td>
</tr>
<tr>
<td>export generator big vessels</td>
<td>0 containers</td>
</tr>
<tr>
<td>rgv full speed</td>
<td>3 m/s</td>
</tr>
<tr>
<td>rgv low speed passing a shortcut</td>
<td>2 m/s</td>
</tr>
<tr>
<td>rgv node speed passing a node</td>
<td>1 m/s</td>
</tr>
<tr>
<td>agv acceleration</td>
<td>1 m/s^2</td>
</tr>
<tr>
<td>minimum distance between trolleys</td>
<td>3 m</td>
</tr>
<tr>
<td>number of stacking cranes at terminal</td>
<td>70 -</td>
</tr>
<tr>
<td>number of agv's per portainer used:</td>
<td></td>
</tr>
<tr>
<td>- unloading a vessel</td>
<td>6 -</td>
</tr>
<tr>
<td>- loading a vessel from different berth</td>
<td>10 -</td>
</tr>
<tr>
<td>- shuffling at the terminal</td>
<td>10 -</td>
</tr>
<tr>
<td>stacking crane cycle time</td>
<td>80 seconds</td>
</tr>
<tr>
<td>mean portainer cycle time</td>
<td>58 seconds</td>
</tr>
</tbody>
</table>

Table 15.1 Input parameters for experiment, shuffling from Berth 1

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>fromtlob[1,2]</td>
<td>0 containers</td>
</tr>
<tr>
<td>fromtlob[3,2]</td>
<td>0 containers</td>
</tr>
<tr>
<td>fromtlob[2,1]</td>
<td>350 containers</td>
</tr>
<tr>
<td>fromtlob[2,3]</td>
<td>0 containers</td>
</tr>
<tr>
<td>Inplacecounter 1</td>
<td>1250 containers</td>
</tr>
<tr>
<td>Inplacecounter 2</td>
<td>1240 containers</td>
</tr>
</tbody>
</table>

Table 15.2 Output parameters for experiment shuffling from Berth 1

In Table 15.2 the parameters inplacecounter 1 & inplacecounter 2 indicate 1250+1240 containers in total have been shuffled to the right location.
15.1.1 Warehouse cranes located in Yard 1 while shuffling from Yard 1

The cranes located in Yard 1 will have to digest 2490 containers. 1250 containers to be delivered to Yard 2 while 1240 containers have to be delivered from Yard 2 to Yard 1.

Considering that the number of containers that will be retrieved and delivered form the 20’ stack and the 40’ stacks are the same means that 1245 containers will be handled by both stacks types.

- Required 20’ warehouse crane operational time: $1245 \times 80$ seconds = 99,600 seconds
- Required 40’ warehouse crane operational time: $1245 \times 80$ seconds = 99,600 seconds

Number of warehouse cranes in Yard 2:
- Number of warehouse cranes in the 20’ stacks in Yard 1 = $4 \times 2 = 8$ warehouse cranes
- Number of warehouse cranes in the 40’ stacks in Yard 2 = $4 \times 3 = 12$ warehouse cranes

Needed operational time per warehouse crane:

- $\frac{99,600}{8} = 12,450$ seconds per 20’ stack warehouse crane
- $\frac{99,600}{12} = 8,300$ seconds per 40’ stack warehouse crane

Required adjacent loading time for both feeder vessels: 7.4 hours = 26,640 seconds

Occupation rates of the warehouse cranes:

20’ stack: $\left(\frac{12,450}{26,640}\right) \times 100\% = 46.7\%$

40’ stack: $\left(\frac{8,300}{26,640}\right) \times 100\% = 31.2\%$
Figure 15.1, queue at node 116 when shuffling from Berth 1

Figure 15.2, queue at node 114 when shuffling from Berth 1

Figure 15.3, queue at node 260. when shuffling from Berth 1

Figure 15.4, queue at node 290 when shuffling from Berth 1
Figure 15.5, queue at node 243 when shuffling from Berth 1

Figure 15.6, queue at node 273 when shuffling from Berth 1
Appendix 16: Input parameters for the experiment with one mega vessel at Berth 2 unloading

The basic parameters used for this experiment are shown in Table 16.1. The changed parameters after the run are shown in Table 16.2.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>fromtob[1,2]</td>
<td>0</td>
</tr>
<tr>
<td>fromtob[3,2]</td>
<td>0</td>
</tr>
<tr>
<td>fromtob[2,1]</td>
<td>0</td>
</tr>
<tr>
<td>fromtob[2,3]</td>
<td>0</td>
</tr>
<tr>
<td>inplacecounter 1</td>
<td>0</td>
</tr>
<tr>
<td>inplacecounter 2</td>
<td>0</td>
</tr>
<tr>
<td>inplacecounter 3</td>
<td>0</td>
</tr>
<tr>
<td>seed of uniform</td>
<td>67</td>
</tr>
<tr>
<td>seed of arrival times feeders</td>
<td>17</td>
</tr>
<tr>
<td>seed of arrival times big</td>
<td>27</td>
</tr>
<tr>
<td>import generator feeders</td>
<td>0</td>
</tr>
<tr>
<td>export generator feeders</td>
<td>0</td>
</tr>
<tr>
<td>import generator big vessels</td>
<td>3150</td>
</tr>
<tr>
<td>export generator big vessels</td>
<td>3150</td>
</tr>
<tr>
<td>rgv full speed</td>
<td>3</td>
</tr>
<tr>
<td>rgv low speed passing a shortcut</td>
<td>2</td>
</tr>
<tr>
<td>rgv node speed passing a node</td>
<td>1</td>
</tr>
<tr>
<td>rgv acceleration</td>
<td>1</td>
</tr>
<tr>
<td>minimum distance between trolleys</td>
<td>3</td>
</tr>
<tr>
<td>number of stacking cranes at terminal</td>
<td>70</td>
</tr>
<tr>
<td>number of agv's per portainer used:</td>
<td>-</td>
</tr>
<tr>
<td>- unloading a vessel</td>
<td>6</td>
</tr>
<tr>
<td>- loading a vessel from different berth</td>
<td>10</td>
</tr>
<tr>
<td>- shuffling at the terminal</td>
<td>10</td>
</tr>
<tr>
<td>stacking crane cycle time</td>
<td>80</td>
</tr>
<tr>
<td>mean portainer cycle time</td>
<td>58</td>
</tr>
<tr>
<td></td>
<td>m/s</td>
</tr>
<tr>
<td></td>
<td>m/s</td>
</tr>
<tr>
<td></td>
<td>m/s²</td>
</tr>
<tr>
<td></td>
<td>m</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>seconds</td>
</tr>
<tr>
<td></td>
<td>seconds</td>
</tr>
</tbody>
</table>

Table 16.1: Input parameters for experiment: Mega vessel at Berth 2 unloading

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>fromtob[2,1]</td>
<td>1579</td>
</tr>
<tr>
<td>fromtob[2,3]</td>
<td>1571</td>
</tr>
</tbody>
</table>

Table 16.2, output after experiment: Mega vessel at Berth 2 unloading

In Table 16.2 the parameters fromtob[2,1] and fromtob[2,3] indicate 1579 containers are unloaded eventually destined for Yard 1 and 1571 containers are unloaded destined for Yard 3.
16.1.1 Crane occupancy rates when unloading a mega vessel

A mega vessel moored at Berth 2 will have its containers unloaded in Yard 2. The number of unloaded containers is 3150. Considering that the number of containers that have been delivered to the 20' stacks and the 40' stacks are the same means that 1575 containers will be delivered to both stack types.

- Required 20' warehouse crane operational time: $1575 \times 80 \text{ seconds} = 126,000 \text{ seconds}$
- Required 40' warehouse crane operational time: $1575 \times 80 \text{ seconds} = 126,000 \text{ seconds}$

Number of warehouse cranes in Yard 2:
- Number of warehouse cranes in the 20' stacks in Yard 2 = $6 \times 2 = 12$ warehouse cranes
- Number of warehouse cranes in the 40' stacks in Yard 2 = $6 \times 3 = 18$ warehouse cranes

Needed operational time per warehouse crane:

- $\frac{126,000}{12} = 10,500 \text{ seconds per 20' stack warehouse crane}$
- $\frac{126,000}{18} = 7,000 \text{ seconds per 40' stack warehouse crane}$

Required unloading time for the mega vessel: $8.9 \text{ hours} \Rightarrow 32,040 \text{ seconds}$

Occupation rates of the warehouse cranes:

20' stack: $\left(\frac{10,500}{32,040}\right) \times 100\% = 32.8\%$

40' stack: $\left(\frac{7,000}{32,040}\right) \times 100\% = 21.8\%$
Appendix 16

Figure 16.1, portainer 5 queue at Berth 2 when unloading and loading a mega vessel

Figure 16.2, portainer 6 queue at Berth 2 when unloading and loading a mega vessel

Figure 16.3, portainer 7 queue at Berth 2 when unloading and loading a mega vessel

Figure 16.4, portainer 8 queue at Berth 2 when unloading and loading a mega vessel
**Appendix 16**

**Figure 16.5, portainer 9 queue at Berth 2 when unloading and loading a mega vessel**

**Figure 16.6, portainer 10 queue at Berth 2 when unloading and loading a mega vessel**

**Figure 16.7, rgv queues on warehouse lane 62 when loading and unloading a mega vessel**

**Figure 16.8, rgv queues on warehouse lane 64 when loading and unloading a mega vessel**
Appendix 16

Figure 16.9, rgv queues on warehouse lane 66 when loading and unloading a mega vessel

Figure 16.10, rgv queues on warehouse lane 68 when loading and unloading a mega vessel

Figure 16.11, rgv queues on warehouse lane 70 when loading and unloading a mega vessel

Figure 16.12, rgv queues on warehouse lane 71 when loading and unloading a mega vessel

63
Figure 16.13, rgv queues on warehouse lane 73 when loading and unloading a mega vessel

Figure 16.14, rgv queues on warehouse lane 75 when loading and unloading a mega vessel

Figure 16.15, rgv queues on warehouse lane 77 when loading and unloading a mega vessel

Figure 16.16, rgv queues on warehouse lane 79 when loading and unloading a mega vessel
Appendix 17: Input parameters for the experiment with one mega vessel at Berth 2, loading

The basic parameters used for the experiment with the mega vessel at Berth 2 when it is loading are shown in Table 17.1. The parameters after the run are shown in Table 17.2. Inplacecounter 2 refers to the number of containers in yard 2 destined to be taken away by a vessel at berth 2. These containers can be loaded fast. In this experiment there are 10000 containers ready in yard 2 to be taken away.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>fromtob[1,2]</td>
<td>0</td>
</tr>
<tr>
<td>fromtob[3,2]</td>
<td>0</td>
</tr>
<tr>
<td>fromtob[2,1]</td>
<td>0</td>
</tr>
<tr>
<td>fromtob[2,3]</td>
<td>0</td>
</tr>
<tr>
<td>inplacecounter 1</td>
<td>0</td>
</tr>
<tr>
<td>inplacecounter 2</td>
<td>10000</td>
</tr>
<tr>
<td>inplacecounter 3</td>
<td>0</td>
</tr>
<tr>
<td>seed of uniform</td>
<td>67</td>
</tr>
<tr>
<td>seed of arrivaltimes feeders</td>
<td>17</td>
</tr>
<tr>
<td>seed of arrivaltimes big</td>
<td>27</td>
</tr>
<tr>
<td>import generator feeders</td>
<td>0</td>
</tr>
<tr>
<td>export generator feeders</td>
<td>0</td>
</tr>
<tr>
<td>import generator big vessels</td>
<td>3150</td>
</tr>
<tr>
<td>export generator big vessels</td>
<td>3150</td>
</tr>
<tr>
<td>rgv full speed</td>
<td>3</td>
</tr>
<tr>
<td>rgv low speed passing a shortcut</td>
<td>2</td>
</tr>
<tr>
<td>rgv node speed passing a node</td>
<td>1</td>
</tr>
<tr>
<td>rgv acceleration</td>
<td>1</td>
</tr>
<tr>
<td>minimum distance between trolleys</td>
<td>3</td>
</tr>
<tr>
<td>number of stacking cranes at terminal</td>
<td>70</td>
</tr>
</tbody>
</table>

- unloading a vessel               | 6         |
- loading a vessel from different berth | 10     |
- shuffling at the terminal         | 10        |
stacking crane cycle time          | 80         |
mean portainer cycle time          | 58         |

Table 17.1: Input parameters for experiment: Mega vessel at berth 2 loading

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>fromtob[2,1]</td>
<td>1571</td>
</tr>
<tr>
<td>fromtob[2,3]</td>
<td>1579</td>
</tr>
<tr>
<td>Inplacecounter 2</td>
<td>6850</td>
</tr>
</tbody>
</table>

Table 17.2, output after experiment: Mega vessel at berth 2 loading

In Table 17.2 the parameters fromtob[2,1] and fromtob[2,3] indicate 1579 containers are unloaded destined for Berth 1 and 1571 containers are unloaded destined for Yard 3. Inplacecounter 2 is 6850 this implies 3150 containers have been taken away by the vessel.
17.1.1 Crane occupancy rates when loading a mega vessel

A mega vessel moored at Berth 2 will have its containers fast loaded from Yard 2. The number of to be loaded containers is 3150. Considering that the number of containers that have have retrieved from the 20' stacks and the 40' stacks are the same means that 1575 containers will be delivered to both stack types.

- Required 20' warehouse crane operational time: $1575 \times 80 \text{ seconds} = 126,000 \text{ seconds}$
- Required 40' warehouse crane operational time: $1575 \times 80 \text{ seconds} = 126,000 \text{ seconds}$

Number of warehouse cranes in Yard 2:
- Number of warehouse cranes in the 20' stacks in Yard 2 $= 6 \times 2 = 12$ warehouse cranes
- Number of warehouse cranes in the 40' stacks in Yard 2 $= 6 \times 3 = 18$ warehouse cranes

Needed operational time per warehouse crane:

- $\frac{126,000}{12} = 10,500 \text{ seconds per 20' stack warehouse crane}$
- $\frac{126,000}{18} = 7,000 \text{ seconds per 40' stack warehouse crane}$

Required unloading time for the mega vessel: 8.9 hours $\Rightarrow 32,040 \text{ seconds}$

Occupation rates of the warehouse cranes:

20' stack: $\left( \frac{10,500}{32,040} \right) \times 100\% = 32.8\%$

40' stack: $\left( \frac{7,000}{32,040} \right) \times 100\% = 21.8\%$
Appendix 18: Input parameters for the experiment with one mega vessel at Berth 2 unloading and loading containers from an adjacent berth

The basic parameters used for the experiment with the mega vessel at Berth 2 when it is being loaded from two adjacent berths are shown in Table 18.1. The parameter fromtobot[1,2] indicates the number of containers in Yard 1 which can be used to load the vessel at Berth 2. The same principle applies for fromtobot[3,2], fromtobot[2,1] and fromtobot[2,3]. Inplacecounter 1 indicates the number of containers in Yard 1 that can be used to load a vessel at Berth 1. In this experiment Inplacecounter 2 is 0, indicating the vessel at Berth 2 will have to be loaded with containers from Yard 1 and Yard 3. In these yards more than enough containers destined for berth 2 have been placed. The changed parameters after the run are shown in Table 18.2.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>fromtobot[1,2]</td>
<td>10000</td>
</tr>
<tr>
<td>fromtobot[3,2]</td>
<td>10000</td>
</tr>
<tr>
<td>fromtobot[2,1]</td>
<td>0</td>
</tr>
<tr>
<td>fromtobot[2,3]</td>
<td>0</td>
</tr>
<tr>
<td>inplacecounter 1</td>
<td>0</td>
</tr>
<tr>
<td>inplacecounter 2</td>
<td>0</td>
</tr>
<tr>
<td>inplacecounter 3</td>
<td>0</td>
</tr>
<tr>
<td>seed of uniform</td>
<td>67</td>
</tr>
<tr>
<td>seed of arrivaltimes feeders</td>
<td>17</td>
</tr>
<tr>
<td>seed of arrivaltimes big</td>
<td>27</td>
</tr>
<tr>
<td>import generator feeders</td>
<td>0</td>
</tr>
<tr>
<td>export generator feeders</td>
<td>0</td>
</tr>
<tr>
<td>import generator big vessels</td>
<td>3150</td>
</tr>
<tr>
<td>export generator big vessels</td>
<td>3150</td>
</tr>
<tr>
<td>rgy full speed</td>
<td>3 m/s</td>
</tr>
<tr>
<td>rgy low speed passing a shortcut</td>
<td>2 m/s</td>
</tr>
<tr>
<td>rgy node speed passing a node</td>
<td>1 m/s</td>
</tr>
<tr>
<td>rgy acceleration</td>
<td>1 m/s²</td>
</tr>
<tr>
<td>minimum distance between trolleys</td>
<td>3 m</td>
</tr>
<tr>
<td>number of stacking cranes at terminal</td>
<td>70</td>
</tr>
<tr>
<td>number of agv's per portainer used:</td>
<td></td>
</tr>
<tr>
<td>- unloading a vessel</td>
<td>6</td>
</tr>
<tr>
<td>- loading a vessel from different berth</td>
<td>10</td>
</tr>
<tr>
<td>- shuffling at the terminal</td>
<td>10</td>
</tr>
<tr>
<td>stacking crane cycle time</td>
<td>80 seconds</td>
</tr>
<tr>
<td>mean portainer cycle time</td>
<td>58 seconds</td>
</tr>
</tbody>
</table>

Table 18.1: Input parameters for experiment: Mega vessel at Berth 2 loading from an adjacent berth

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>fromtobot[1,2]</td>
<td>8453</td>
</tr>
<tr>
<td>fromtobot[3,2]</td>
<td>8397</td>
</tr>
<tr>
<td>fromtobot[2,1]</td>
<td>1579</td>
</tr>
<tr>
<td>fromtobot[2,3]</td>
<td>1571</td>
</tr>
</tbody>
</table>

Table 18.2, output after experiment: Mega vessel at Berth 2 loading from an adjacent berth

In Table 18.2 fromtobot[2,1] is 8453. This means that from the original 10000 containers in Yard 1 destined for a vessel at Berth 2, 8453 containers are left in Yard 1, indicating that: 10000-8453=1547 containers from Yard 1 are used to load the mega vessel at Berth 2.
18.1.1 Crane occupancy rates

A megavessel moored at Berth 2 will have its containers adjacent loaded from Yard 1 & 3. The number of to be loaded containers is 3150. Considering that the number of containers that have been retrieved from the 20' stacks and the 40' stacks are the same means that 1575 containers will be delivered to both stack types in Yard 1 & 3.

- Required 20' warehouse crane operational time: 1575 * 80 seconds = 126,000 seconds
- Required 40' warehouse crane operational time: 1575 * 80 seconds = 126,000 seconds

Number of warehouse cranes in Yard 1 & 3:
- Number of warehouse cranes in the 20' stacks = 8 * 2 = 16 warehouse cranes
- Number of warehouse cranes in the 40' stacks = 8 * 3 = 24 warehouse cranes

Needed operational time per warehouse crane:

- \[ \frac{126,000}{16} = 7,875 \text{ seconds per 20' stack warehouse crane} \]
- \[ \frac{126,000}{24} = 5,250 \text{ seconds per 40' stack warehouse crane} \]

Required unloading time for the feeder vessel: 9.7 hours \(\Rightarrow 34,920 \text{ seconds} \)

Occupation rates of the warehouse cranes:

20' stack: \( \left( \frac{7,875}{34,920} \right) \times 100\% = 22.6\% \)

40' stack: \( \left( \frac{5,250}{34,920} \right) \times 100\% = 15.0\% \)
Appendix 18

Figure 18.1, portainer 5 queue when unloading and loading from an adjacent berth at Berth 2

Figure 18.2, portainer 6 queue when unloading and loading from an adjacent berth at Berth 2

Figure 18.3, portainer 7 queue when unloading and loading from an adjacent berth at Berth 2

Figure 18.4, portainer 8 queue when unloading and loading from an adjacent berth at Berth 2
Figure 18.5, portainer 9 queue when unloading and loading from an adjacent berth at Berth 2

Figure 18.6, portainer 10 queue when unloading and loading from an adjacent berth at Berth 2
Figure 18.7, node row at node 243 when adjacent loading at Berth 2

Figure 18.8, node row at node 273 when adjacent loading at Berth 2

Figure 18.9, node row at node 303 when adjacent loading at Berth 2

Figure 18.10, node row at node 363 when adjacent loading at Berth 2
Figure 18.11, node row at node 393 when adjacent loading at Berth 2

Figure 18.12, node row at node 423 when adjacent loading at Berth 2

Figure 18.13, node row at node 453 when adjacent loading at Berth 2

Figure 18.14, node row at node 260 when adjacent loading at Berth 2
Figure 18.15, node row at node 290 when adjacent loading at Berth 2

Figure 18.16, node row at node 320 when adjacent loading at Berth 2

Figure 18.17, node row at node 350 when adjacent loading at Berth 2

Figure 18.18, node row at node 380 when adjacent loading at Berth 2

73
Figure 18.19, node row at node 410 when adjacent loading at Berth 2

Figure 18.20, node row at node 440 when adjacent loading at Berth 2
Appendix 19: Input parameters for the experiment "shuffling from Berth 2"

The basic parameters used for the experiment shuffling from Berth 2 are shown in Table 19.1. The shuffle process stops when there are no more containers in Yard 2 to be taken to Yard 1 or 3. In this process there are always containers available in Yard 1 and Yard 3 while frombtob[1,2] and frombtob[3,2] are 10000 each. The changed parameters after the run are shown in Table 19.2.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>frombtob[1,2]</td>
<td>10000</td>
</tr>
<tr>
<td>frombtob[3,2]</td>
<td>10000</td>
</tr>
<tr>
<td>frombtob[2,1]</td>
<td>1575</td>
</tr>
<tr>
<td>frombtob[2,3]</td>
<td>1575</td>
</tr>
<tr>
<td>inplacecounter 1</td>
<td>0</td>
</tr>
<tr>
<td>inplacecounter 2</td>
<td>0</td>
</tr>
<tr>
<td>inplacecounter 3</td>
<td>0</td>
</tr>
<tr>
<td>seed of uniform</td>
<td>67</td>
</tr>
<tr>
<td>seed of arrivaltimes feeders</td>
<td>17</td>
</tr>
<tr>
<td>seed of arrivaltimes big</td>
<td>27</td>
</tr>
<tr>
<td>import generator feeders</td>
<td>0</td>
</tr>
<tr>
<td>export generator feeders</td>
<td>0</td>
</tr>
<tr>
<td>import generator big vessels</td>
<td>0</td>
</tr>
<tr>
<td>export generator big vessels</td>
<td>0</td>
</tr>
<tr>
<td>rgy full speed</td>
<td>3</td>
</tr>
<tr>
<td>rgy low speed passing a shortcut</td>
<td>2</td>
</tr>
<tr>
<td>rgy node speed passing a node</td>
<td>1</td>
</tr>
<tr>
<td>rgy acceleration</td>
<td>1</td>
</tr>
<tr>
<td>minimum distance between trolleys</td>
<td>3</td>
</tr>
<tr>
<td>number of stacking cranes at terminal</td>
<td>70</td>
</tr>
<tr>
<td>number of agv's per portainer used:</td>
<td></td>
</tr>
<tr>
<td>- unloading a vessel</td>
<td>6</td>
</tr>
<tr>
<td>- loading a vessel from different berth</td>
<td>10</td>
</tr>
<tr>
<td>- shuffling at the terminal</td>
<td>10</td>
</tr>
<tr>
<td>stacking crane cycle time</td>
<td>80</td>
</tr>
<tr>
<td>mean portainer cycle time</td>
<td>58</td>
</tr>
</tbody>
</table>

Table 19.1: Input parameters for experiment: shuffling from Berth 2

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>frombtob[1,2]</td>
<td>8511</td>
</tr>
<tr>
<td>frombtob[3,2]</td>
<td>8439</td>
</tr>
<tr>
<td>frombtob[2,1]</td>
<td>0</td>
</tr>
<tr>
<td>frombtob[2,3]</td>
<td>0</td>
</tr>
<tr>
<td>inplacecounter 1</td>
<td>1533</td>
</tr>
<tr>
<td>inplacecounter 2</td>
<td>3150</td>
</tr>
<tr>
<td>inplacecounter 3</td>
<td>1617</td>
</tr>
</tbody>
</table>

Table 19.2, output after experiment: shuffling from Berth 2

In Table 19.2 the parameters inplacecounter 1, 2 and 3 indicate 6300 containers in total have been shuffled between the yards.
19.1.1 crane occupancy rates

The cranes located in Yard 2 will have to deal with 6300 containers. 3150 containers to be delivered to Yard 1 & 3 from Yard 2, while 3150 containers have to be delivered from Yard 2 to Yard 1 & 3.

Considering that the number of containers that will be retrieved and delivered form the 20’ stack and the 40’ stacks are the same means that 3150 containers will be handled by both stacks types located on Yard 2.

- Required 20’ warehouse crane operational time: \(3150 \times 80 \text{ seconds} = 252,000 \text{ seconds}\)
- Required 40’ warehouse crane operational time: \(3150 \times 80 \text{ seconds} = 252,000 \text{ seconds}\)

Number of warehouse cranes in Yard 2:
- Number of warehouse cranes in the 20’ stacks in Yard 2 = 6 \times 2 = 12 warehouse cranes
- Number of warehouse cranes in the 40’ stacks in Yard 2 = 6 \times 3 = 18 warehouse cranes

Needed operational time per warehouse crane:

- \(\frac{252,000}{12} = 21,000 \text{ seconds per 20’ stack warehouse crane}\)
- \(\frac{252,000}{18} = 14,000 \text{ seconds per 40’ stack warehouse crane}\)

Required adjacent loading time for both feeder vessels: 11.25 hours \(\Rightarrow 40,500 \text{ seconds}\)

Occupation rates of the warehouse cranes:

- 20’ stack: \(\left(\frac{21,000}{40,500}\right) \times 100\% = 51.9\%\)
- 40’ stack: \(\left(\frac{14,000}{40,500}\right) \times 100\% = 34.6\%\)
Figure 19.1, rgv queues in warehouse lane 21 when shuffling from Berth 2

Figure 19.2, rgv queues in warehouse lane 22 when shuffling from Berth 2

Figure 19.3, rgv queues in warehouse lane 23 when shuffling from Berth 2

Figure 19.4, rgv queues in warehouse lane 24 when shuffling from Berth 2
Figure 19.5, rgv queues in warehouse lane 25 when shuffling from Berth 2

Figure 19.6, rgv queues in warehouse lane 26 when shuffling from Berth 2

Figure 19.7, rgv queues in warehouse lane 27 when shuffling from Berth 2

Figure 19.8, rgv queues in warehouse lane 28 when shuffling from Berth 2
Appendix 19

Figure 19.9, rgv queues in warehouse lane 29 when shuffling from Berth 2

Figure 19.10, rgv queues in warehouse lane 30 when shuffling from Berth 2

Figure 19.11, rgv waiting queue at node 260

Figure 19.12, rgv waiting queue at node 290
Figure 19.17, rgv waiting queue at node 440

Figure 19.18, rgv waiting queue at node 243

Figure 19.19, rgv waiting queue at node 273

Figure 19.20, rgv waiting queue at node 303
Figure 19.21, rgv waiting queue at node 333

Figure 19.22, rgv waiting queue at node 363

Figure 19.23, rgv waiting queue at node 393

Figure 19.24, rgv waiting queue at node 423
Appendix 20: Input parameters for the experiment adjacent loading at all three berths

The basic parameters used for the experiment adjacent loading from all three berths are shown in Table 20.1. The import generators of the feeder and the mega vessel only produce a few containers while this is not important in this experiment. The parameters fromtob[1,2] represent the containers in Yard 1 which have to be loaded unto a vessel at Berth 2. The changed parameters after the run are shown in Table 20.2.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>fromtob[1,2]</td>
<td>2500 containers</td>
</tr>
<tr>
<td>fromtob[3,2]</td>
<td>2500 containers</td>
</tr>
<tr>
<td>fromtob[2,1]</td>
<td>1500 containers</td>
</tr>
<tr>
<td>fromtob[2,3]</td>
<td>1500 containers</td>
</tr>
<tr>
<td>inplacecounter 1</td>
<td>0 containers</td>
</tr>
<tr>
<td>inplacecounter 2</td>
<td>0 containers</td>
</tr>
<tr>
<td>inplacecounter 3</td>
<td>0 containers</td>
</tr>
<tr>
<td>seed of uniform</td>
<td>67 -</td>
</tr>
<tr>
<td>seed of arrivaltimes feeders</td>
<td>17 -</td>
</tr>
<tr>
<td>seed of arrivaltimes big</td>
<td>27 -</td>
</tr>
<tr>
<td>import generator feeders</td>
<td>80 containers</td>
</tr>
<tr>
<td>export generator feeders</td>
<td>1050 containers</td>
</tr>
<tr>
<td>import generator big vessels</td>
<td>120 containers</td>
</tr>
<tr>
<td>export generator big vessels</td>
<td>3150 containers</td>
</tr>
<tr>
<td>rgv full speed</td>
<td>3 m/s</td>
</tr>
<tr>
<td>rgv low speed passing a shortcut</td>
<td>2 m/s</td>
</tr>
<tr>
<td>rgv node speed passing a node</td>
<td>1 m/s</td>
</tr>
<tr>
<td>rgv acceleration</td>
<td>1 m/s²</td>
</tr>
<tr>
<td>minimum distance between trolleys</td>
<td>3 m</td>
</tr>
<tr>
<td>number of stacking cranes at terminal</td>
<td>70 -</td>
</tr>
<tr>
<td>number of agv's per portainer used:</td>
<td>-</td>
</tr>
<tr>
<td>- unloading a vessel</td>
<td>6 -</td>
</tr>
<tr>
<td>- loading a vessel from different berth</td>
<td>10 -</td>
</tr>
<tr>
<td>- shuffling at the terminal</td>
<td>10 -</td>
</tr>
<tr>
<td>stacking crane cycle time</td>
<td>80 seconds</td>
</tr>
<tr>
<td>mean portainer cycle time</td>
<td>58 seconds</td>
</tr>
</tbody>
</table>

Table 20.1: Input parameters for experiment: adjacent loading at all three berths

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>fromtob[1,2]</td>
<td>1018 containers</td>
</tr>
<tr>
<td>fromtob[3,2]</td>
<td>912 containers</td>
</tr>
<tr>
<td>fromtob[2,1]</td>
<td>485 containers</td>
</tr>
<tr>
<td>fromtob[2,3]</td>
<td>475 containers</td>
</tr>
</tbody>
</table>

Table 20.2, output after experiment adjacent loading at all 3 berths
20.1.1 Warehouse cranes located in Yard 2 while adjacent loading on all three the berths

As both feeder vessels will be provided from Yard 2 this means that $1050 \times 2 = 2100$ containers have to retrieved from these stacks. Considering that the number of containers that will retrieved from the 20' stacks and the 40' stacks are the same means that 1050 will be retrieved from both stack types.

Required 20' warehouse crane operational time: $1050 \times 80$ seconds = 84,000 seconds
Required 40' warehouse crane operational time: $1050 \times 80$ seconds = 84,000 seconds

Number of warehouse cranes in Yard 2:
Number of warehouse cranes in the 20' stacks in Yard 2 = $6 \times 2 = 12$ warehouse cranes
Number of warehouse cranes in the 40' stacks in Yard 2 = $6 \times 3 = 18$ warehouse cranes

Needed operational time per warehouse crane:

$$\frac{84,000}{12} = 7,000 \text{ seconds per 20' stack warehouse crane}$$
$$\frac{84,000}{18} = 4,667 \text{ seconds per 40' stack warehouse crane}$$

Required adjacent loading time for both feeder vessels: 5.4 hours $\Rightarrow$ 20160 seconds

Occupation rates of the warehouse cranes:

20' stack: $\left(\frac{7,000}{20160}\right) \times 100\% = 34.7\%$

40' stack: $\left(\frac{4,667}{20160}\right) \times 100\% = 23.2\%$

In the same manner the occupancy rates of the warehouse cranes located in Yard 1 and 3 have been calculated. In 5.4 hours 1752 containers have been handled.

Required 20' warehouse crane operational time: $876 \times 80$ seconds = 70,080 seconds
Required 40' warehouse crane operational time: $876 \times 80$ seconds = 70,080 seconds

Occupation rates of the warehouse cranes:

20' stack: $\left(\frac{8,760}{20,160}\right) \times 100\% = 43.5\%$

40' stack: $\left(\frac{5,840}{20,160}\right) \times 100\% = 29.0\%$
Appendix 20

Figure 20.1 rgv queue under portainer 1

Figure 20.2 rgv queue under portainer 2

Figure 20.3 rgv queue under portainer 3

Figure 20.4, rgv queue under portainer 4
Appendix 20

Figure 20.5 rgv queue under portainer 5

Figure 20.6 rgv queue under portainer 7

Figure 20.7 rgv queue under portainer 7

Figure 20.8 rgv queue under portainer 8
Appendix 20

Figure 20.9 rgv queue under portainer 9

Figure 20.10 rgv queue under portainer 10

Figure 20.11 rgv queue under portainer 11

Figure 20.12 rgv queue under portainer 12
Appendix 20

Figure 20.13 rgv queue under portainer 13

Figure 20.14 rgv queue under portainer 14
Appendix 20

Figure 20.15 rgv queue at node 110

Figure 20.16 rgv queue at node 350

Figure 20.17 rgv queue at node 108

Figure 20.18 rgv queue at node 380

89
Appendix 20

Figure 20.19 rgv queue at node 106

Figure 20.20 rgv queue at node 508

Figure 20.21 rgv queue at node 333

Figure 20.22, rgv queue at node 510
Figure 20.27 rgv queue at node 85

Figure 20.28 rgv queue at node 73

Figure 20.29 rgv queue at node 86

Figure 20.30 rgv queue at node 113
Appendix 20

Figure 20.31 rgv queue at node 69

Figure 20.32 rgv queue at node 90

Figure 20.33 rgv queue at node 109
Appendix 20

Figure 20.34 rgv queue at node 51

Figure 20.35 rgv queue at node 68

Figure 20.36 rgv queue at node 91

Figure 20.37 rgv queue at node 96
Figure 20.38 rgv queue at node 62

Figure 20.39 rgv queue at node

Figure 20.40 rgv queue at node 535

Figure 20.41 rgv queue at node 534
Figure 20.42 rgv queue at node 505

Figure 20.43 rgv queue at node 569

Figure 20.44 rgv queue at node 550

Figure 20.45 rgv queue at node 529
Figure 20.46 rgv queue at node 551

Figure 20.47 rgv queue at node 528

Figure 20.48 rgv queue at node 511

Figure 20.49 rgv queue at node 363
Figure 20.50 rgv queue at node 544

Figure 20.51 rgv queue at node 556

Figure 20.52 rgv queue at node 523

Figure 20.53 rgv queue at node 517
Figure 20.54 rgv queue in warehouse lane 101

Figure 20.55 queue in warehouse lane 102

Figure 20.56 queue in warehouse lane 103

Figure 20.57 queue in warehouse lane 104
Appendix 20

Figure 20.58 queue in warehouse lane 105

Figure 20.59 queue in warehouse lane 106

Figure 20.60 queue in warehouse lane 107

Figure 20.61 queue in warehouse lane 108
Figure 20.62 queue in warehouse lane 109

Figure 20.63 queue in warehouse lane 110
Appendix 21: Input parameters for the experiment adjacent loading at Berth 1 and 3 and unloading at Berth 2.

The basic parameters used for this experiment are shown in Table 21.1. The parameters are the same as used in experiment 10, except for the number of rgy's used when loading a vessel from an adjacent berth. In this experiment that is the case for Berth 1 and Berth 3.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>frombtofb[1,2]</td>
<td>0</td>
</tr>
<tr>
<td>frombtofb[3,2]</td>
<td>0</td>
</tr>
<tr>
<td>frombtofb[2,1]</td>
<td>1600</td>
</tr>
<tr>
<td>frombtofb[2,3]</td>
<td>1600</td>
</tr>
<tr>
<td>inplacecounter 1</td>
<td>0</td>
</tr>
<tr>
<td>inplacecounter 2</td>
<td>0</td>
</tr>
<tr>
<td>inplacecounter 3</td>
<td>0</td>
</tr>
<tr>
<td>seed of uniform</td>
<td>67</td>
</tr>
<tr>
<td>seed of arrivaltimes feeders</td>
<td>17</td>
</tr>
<tr>
<td>seed of arrivaltimes big</td>
<td>27</td>
</tr>
<tr>
<td>import generator feeders</td>
<td>40</td>
</tr>
<tr>
<td>export generator feeders</td>
<td>1050</td>
</tr>
<tr>
<td>import generator big vessels</td>
<td>3150</td>
</tr>
<tr>
<td>export generator big vessels</td>
<td>0</td>
</tr>
<tr>
<td>rgv full speed</td>
<td>3</td>
</tr>
<tr>
<td>rgv low speed passing a shortcut</td>
<td>2</td>
</tr>
<tr>
<td>rgv node speed passing a node</td>
<td>1</td>
</tr>
<tr>
<td>rgv acceleration</td>
<td>1</td>
</tr>
<tr>
<td>minimum distance between trolleys</td>
<td>3</td>
</tr>
<tr>
<td>number of stacking cranes at terminal</td>
<td>70</td>
</tr>
<tr>
<td>number of agv's per portainer used:</td>
<td></td>
</tr>
<tr>
<td>- unloading a vessel</td>
<td>6</td>
</tr>
<tr>
<td>- loading a vessel from different berth</td>
<td>10</td>
</tr>
<tr>
<td>- shuffling at the terminal</td>
<td>10</td>
</tr>
<tr>
<td>stacking crane cycle time</td>
<td>80</td>
</tr>
<tr>
<td>mean portainer cycle time</td>
<td>58</td>
</tr>
</tbody>
</table>

*Table 21.1 input parameters for experiment; Adjacent loading at Berth 1 and Berth 3 and unloading at Berth 2.*

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>frombtofb[1,2]</td>
<td>40</td>
</tr>
<tr>
<td>frombtofb[3,2]</td>
<td>40</td>
</tr>
<tr>
<td>frombtofb[2,1]</td>
<td>2083</td>
</tr>
<tr>
<td>frombtofb[2,3]</td>
<td>2087</td>
</tr>
</tbody>
</table>

*Table 21.2 output parameter values for experiment; Adjacent loading at Berth 1 and Berth 3 and unloading at Berth 2.*
21.1.1 Crane occupancy rates when adjacent loading on Berths 1&3 and unloading on Berth 2

As both feeder vessels will be provided from Yard 2 this means that 1050 * 2 = 2100 containers have to retrieved from these stacks. Both feeder vessels will leave the terminal after 5.3 hours. In this time span the mega vessel has unloaded 1960 containers into Yard 2. Adding these up to the 2100 containers loaded onto both feeder vessels will amount to 4060 containers.

Considering that the number of containers that will retrieved from the 20’ stacks and the 40’ stacks are the same means that 2030 containers will be retrieved from both stack types.

- Required 20’ warehouse crane operational time: 2030 * 80 seconds = 162,400 seconds
- Required 40’ warehouse crane operational time: 2030 * 80 seconds = 162,400 seconds

Number of warehouse cranes in Yard 2:
- Number of warehouse cranes in the 20’ stacks in Yard 2 = 6 * 2 = 12 warehouse cranes
- Number of warehouse cranes in the 40’ stacks in Yard 2 = 6 * 3 = 18 warehouse cranes

Needed operational time per warehouse crane:

- \[ \frac{162,400}{12} = 13,533 \text{ seconds per 20’ stack warehouse crane} \]
- \[ \frac{162,400}{18} = 9,022 \text{ seconds per 40’ stack warehouse crane} \]

Required adjacent loading time for both feeder vessels: 5.3 hours \( \Rightarrow \) 19,080 seconds

Occupation rates of the warehouse cranes:

20’ stack: \( \left( \frac{13,533}{19,080} \right) \times 100\% = 70.1\% \)

40’ stack: \( \left( \frac{9,022}{19,080} \right) \times 100\% = 47.3\% \)
Appendix 21

Figure 21.1 Portainer 1 queue at Berth 1 when adjacent loading a feeder

Figure 21.2 Portainer 2 queue at Berth 1 when adjacent loading a feeder

Figure 21.3 Portainer 3 queue at Berth 1 when adjacent loading a feeder

Figure 21.4 Portainer 4 queue at Berth 1 when adjacent loading a feeder
Figure 21.5 Portainer 5 queue at Berth 2 when unloading a mega vessel

Figure 21.6 Portainer 6 queue at Berth 2 when unloading a mega vessel

Figure 21.7 Portainer 7 queue at Berth 2 when unloading a mega vessel

Figure 21.8 Portainer 8 queue at Berth 2 when unloading a mega vessel
Appendix 21

Figure 21.9 Portainer 9 queue at Berth 2 when unloading a mega vessel

Figure 21.10 Portainer 10 queue at Berth 2 when unloading a mega vessel

Figure 21.11 Portainer 11 queue at Berth 3 when adjacent loading a feeder

Figure 21.12 Portainer 12 queue at Berth 3 when adjacent loading a feeder
Appendix 21

Portainer 13 queue

Figure 21.13 Portainer 13 queue at Berth 3 when adjacent loading a feeder

Portainer 14 queue

Figure 21.14 Portainer 14 queue at Berth 3 when adjacent loading a feeder
Figure 21.19 rgv queue on warehouse lane 104

Figure 21.20 rgv queue on warehouse lane 105

Figure 21.21 rgv queue on warehouse lane 106

Figure 21.22 rgv queue on warehouse lane 107
Appendix 21

Figure 21.23 rgv queue on warehouse lane 108

Figure 21.24 rgv queue on warehouse lane 109

Figure 21.25 rgv queue on warehouse lane 120
Appendix 21

Figure 21.26 node row at node 303

Figure 21.27 node row at node 333

Figure 21.28 node row at node 363

Figure 21.29 node row at node 393
Appendix 21

Figure 21.30 Node row at node 506

Figure 21.31 Node row at node 508

Figure 21.32 Node row at node 510

Figure 21.33 Node row at node 512

113
Figure 21.34 node row at node 350

Figure 21.35 node row at node 380
Appendix 22: Input parameters for the experiment adjacent loading on Berth 2, unloading on Berths 1 & 3

The basic parameters used for this experiment are shown in Table 22.1. The parameter values after the run are shown in Table 22.2.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>frombtob[1,2]</td>
<td>2000</td>
<td>containers</td>
</tr>
<tr>
<td>frombtob[3,2]</td>
<td>2000</td>
<td>containers</td>
</tr>
<tr>
<td>frombtob[2,1]</td>
<td>0</td>
<td>containers</td>
</tr>
<tr>
<td>frombtob[2,3]</td>
<td>0</td>
<td>containers</td>
</tr>
<tr>
<td>inplacecounter 1</td>
<td>50</td>
<td>containers</td>
</tr>
<tr>
<td>inplacecounter 2</td>
<td>0</td>
<td>containers</td>
</tr>
<tr>
<td>inplacecounter 3</td>
<td>50</td>
<td>containers</td>
</tr>
<tr>
<td>seed of uniform</td>
<td>67</td>
<td></td>
</tr>
<tr>
<td>seed of arrivaltimes feeders</td>
<td>17</td>
<td></td>
</tr>
<tr>
<td>seed of arrivaltimes big</td>
<td>27</td>
<td></td>
</tr>
<tr>
<td>import generator feeders</td>
<td>1050</td>
<td>containers</td>
</tr>
<tr>
<td>export generator feeders</td>
<td>40</td>
<td>containers</td>
</tr>
<tr>
<td>import generator big vessels</td>
<td>60</td>
<td>containers</td>
</tr>
<tr>
<td>export generator big vessels</td>
<td>3150</td>
<td>containers</td>
</tr>
<tr>
<td>rgv full speed</td>
<td>3</td>
<td>m/s</td>
</tr>
<tr>
<td>rgv low speed passing a shortcut</td>
<td>2</td>
<td>m/s</td>
</tr>
<tr>
<td>rgv node speed passing a node</td>
<td>1</td>
<td>m/s</td>
</tr>
<tr>
<td>aqv acceleration</td>
<td>1</td>
<td>m/s²</td>
</tr>
<tr>
<td>minimum distance between trolleys</td>
<td>3</td>
<td>m</td>
</tr>
<tr>
<td>number of stacking cranes at terminal</td>
<td>70</td>
<td></td>
</tr>
<tr>
<td>number of aqv's per portainer used:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>- unloading a vessel</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>- loading a vessel from different berth</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>- shuffling at the terminal</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>stacking crane cycle time</td>
<td>80</td>
<td>seconds</td>
</tr>
<tr>
<td>mean portainer cycle time</td>
<td>58</td>
<td>seconds</td>
</tr>
</tbody>
</table>

*Table 22.1: Input parameters for experiment, adjacent loading on Berth 2, unloading on Berth 1 & Berth 3*

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>frombtob[1,2]</td>
<td>1510</td>
<td>containers</td>
</tr>
<tr>
<td>frombtob[3,2]</td>
<td>1440</td>
<td>containers</td>
</tr>
<tr>
<td>frombtob[2,1]</td>
<td>34</td>
<td>containers</td>
</tr>
<tr>
<td>frombtob[2,3]</td>
<td>26</td>
<td>containers</td>
</tr>
<tr>
<td>inplacecounter 1</td>
<td>10</td>
<td>containers</td>
</tr>
<tr>
<td>inplacecounter 2</td>
<td>0</td>
<td>containers</td>
</tr>
<tr>
<td>inplacecounter 3</td>
<td>10</td>
<td>containers</td>
</tr>
</tbody>
</table>

*Table 22.2, output after experiment, adjacent loading on Berth 2, unloading on Berth 1*

115
22.1.1 Situation 11: Adjacent loading on Berth 2; unloading on Berths 1 & 3

As the feeder vessels will unload in Yards 1 & 3 this means that 1050 containers have to be deposited in these stacks. The rgv's adjacent loading for Berth 2 will also retrieve their containers from Yards 1 & 3, bringing the total of handled containers to 3150/2 + 1050 = 2125.

Considering that the number of containers that will retrieved from the 20' stacks and the 40' stacks are the same means that 1063 containers will be retrieved from both stack types.

- Required 20' warehouse crane operational time: 1063 * 80 seconds = 85,040 seconds
- Required 40' warehouse crane operational time: 1063 * 80 seconds = 85,040 seconds

Number of warehouse cranes in Yard 2:
- Number of warehouse cranes in the 20' stacks in Yard 1 = 4 * 2 = 8 warehouse cranes
- Number of warehouse cranes in the 40' stacks in Yard 1 = 4 * 3 = 12 warehouse cranes

Needed operational time per warehouse crane:

- \( \frac{85,040}{8} = 10,630 \) seconds per 20' stack warehouse crane
- \( \frac{85,040}{12} = 7,090 \) seconds per 40' stack warehouse crane

Required adjacent loading time for both feeder vessels: 4.5 hours \( \Rightarrow 16,200 \) seconds

Occupation rates of the warehouse cranes:

20' stack: \( \left( \frac{10,630}{16,200} \right) \times 100\% = 65.6\% \)

40' stack: \( \left( \frac{7,090}{16,200} \right) \times 100\% = 43.7\% \)
Appendix 22

Figure 22.1, rgv row under portainer 1

Figure 22.2, rgv row under portainer 2

Figure 22.3, rgv row under portainer 3

Figure 22.4, rgv row under portainer 4
Appendix 22

Figure 22.9, rgv row on warehouse lane 31

Figure 22.10, rgv row on warehouse lane 32

Figure 22.11, rgv row on warehouse lane 33

Figure 22.12, rgv row on warehouse lane 34
Figure 22.13, rgv row on warehouse lane 35

Figure 22.14, rgv row on warehouse lane 36

Figure 22.15, rgv row on warehouse lane 37

Figure 22.16, rgv row on warehouse lane 38
Appendix 22

Figure 22.17, rgv row on warehouse lane 39

Figure 22.18, rgv row on warehouse lane 40

Figure 22.19, rgv row on node 114

Figure 22.20, rgv row on node 243
Figure 22.25, rgv row on node 506
Appendix 23: Input parameters for the experiment "shuffling from 3 berths"

The basic parameters used for this experiment are shown in Table 23.1. The parameters fromtob represent the containers that need to be shuffled between the yards. Inplacecounter 1, 2 and 3 contain some containers. These are to load the three vessels before the shuffling can start.

The parameter values after the run are shown in Table 23.2.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>fromtob[1,2]</td>
<td>1200</td>
</tr>
<tr>
<td>fromtob[3,2]</td>
<td>1200</td>
</tr>
<tr>
<td>fromtob[2,1]</td>
<td>1600</td>
</tr>
<tr>
<td>fromtob[2,3]</td>
<td>1600</td>
</tr>
<tr>
<td>inplacecounter 1</td>
<td>50</td>
</tr>
<tr>
<td>inplacecounter 2</td>
<td>70</td>
</tr>
<tr>
<td>inplacecounter 3</td>
<td>50</td>
</tr>
<tr>
<td>seed of uniform</td>
<td>67</td>
</tr>
<tr>
<td>seed of arrivaltimes feeders</td>
<td>17</td>
</tr>
<tr>
<td>seed of arrivaltimes big</td>
<td>27</td>
</tr>
<tr>
<td>import generator feeders</td>
<td>40</td>
</tr>
<tr>
<td>export generator feeders</td>
<td>40</td>
</tr>
<tr>
<td>import generator big vessels</td>
<td>60</td>
</tr>
<tr>
<td>export generator big vessels</td>
<td>60</td>
</tr>
<tr>
<td>rgv full speed</td>
<td>3</td>
</tr>
<tr>
<td>rgv low speed passing a shortcut</td>
<td>2</td>
</tr>
<tr>
<td>rgv node speed passing a node</td>
<td>1</td>
</tr>
<tr>
<td>rgv acceleration</td>
<td>1</td>
</tr>
<tr>
<td>minimum distance between trolleys</td>
<td>3</td>
</tr>
<tr>
<td>number of stacking cranes at terminal</td>
<td>70</td>
</tr>
<tr>
<td>number of agv's per portainer used:</td>
<td></td>
</tr>
<tr>
<td>- unloading a vessel</td>
<td>6</td>
</tr>
<tr>
<td>- loading a vessel from different berth</td>
<td>10</td>
</tr>
<tr>
<td>- shuffling at the terminal</td>
<td>10</td>
</tr>
<tr>
<td>stacking crane cycle time</td>
<td>80</td>
</tr>
<tr>
<td>mean portainer cycle time</td>
<td>58</td>
</tr>
</tbody>
</table>

Table 23.1: Input parameters for experiment, "shuffling from 3 berths"

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>fromtob[1,2]</td>
<td>0</td>
</tr>
<tr>
<td>fromtob[3,2]</td>
<td>0</td>
</tr>
<tr>
<td>fromtob[2,1]</td>
<td>0</td>
</tr>
<tr>
<td>fromtob[2,3]</td>
<td>0</td>
</tr>
<tr>
<td>Inplacecounter 1</td>
<td>1631</td>
</tr>
<tr>
<td>Inplacecounter 2</td>
<td>2490</td>
</tr>
<tr>
<td>Inplacecounter 3</td>
<td>1649</td>
</tr>
</tbody>
</table>

Table 23.2, output after experiment, "Shuffling from 3 berths"

In Table 23.2 the parameters inplacecounter 1, 2 and 3 indicate 5770 containers in total have been shuffled between the berths 1, 2 and 3. Actually only 5740 containers have been shuffled, 10 containers behind each berth already where in place. Fromtob[1,2] =0 indicates there are no containers left in Yard 1 destined to go to Yard 2.
23.1.1 Warehouse cranes located in Yard 2 while shuffling on all three the berths

The cranes located in Yard 2 will have to deal with 5600 containers. From Yard 1 & 3 the rgv’s will deliver $2 \times 1200 = 2400$ containers, while Yard 2 itself must deliver 3200 containers to Yard 1 & 3.

Considering that the number of containers that will be retrieved and delivered form the 20’ stack and the 40’ stacks are the same means that 2800 containers will be handled by both stacks types.

- Required 20’ warehouse crane operational time: $2800 \times 80$ seconds = 224,000 seconds
- Required 40’ warehouse crane operational time: $2800 \times 80$ seconds = 224,000 seconds

Number of warehouse cranes in Yard 2:
- Number of warehouse cranes in the 20’ stacks in Yard 2 = $6 \times 2 = 12$ warehouse cranes
- Number of warehouse cranes in the 40’ stacks in Yard 2 = $6 \times 3 = 18$ warehouse cranes

Needed operational time per warehouse crane:

- \[
\frac{224,000}{12} = 18,667 \text{ seconds per 20'} \text{ stack warehouse crane}
\]
- \[
\frac{224,000}{18} = 12,444 \text{ seconds per 40'} \text{ stack warehouse crane}
\]

Required adjacent loading time for both feeder vessels: 6.5 hours $\Rightarrow 23400$ seconds

Occupation rates of the warehouse cranes:

20’ stack: \[
\left(\frac{18,667}{23,400}\right) \times 100\% = 79.8 \%
\]

40’ stack: \[
\left(\frac{12,444}{23,400}\right) \times 100\% = 53.2 \%
\]
Appendix 23
Figure 23.5, rgv queue at node 106

Figure 23.6, rgv queue at node 508

Figure 23.7, rgv queue at node 333

Figure 23.8, rgv queue at node 510
Appendix 23

Figure 23.9, rgv queue at node 363

Figure 23.10, rgv queue at node 512
Appendix 23

Figure 23.11, RGV queue in warehouse lane 101

Figure 23.12, queue in warehouse lane 102

Figure 23.13, queue in warehouse lane 103

Figure 23.14, queue in warehouse lane 104

130
Figure 23.15, queue in warehouse lane 105

Figure 23.16, queue in warehouse lane 106

Figure 23.17, queue in warehouse lane 107

Figure 23.18, queue in warehouse lane 108
Appendix 23

Figure 23.19, queue in warehouse lane 109

Figure 23.20, queue in warehouse lane 110
Appendix 24: Input parameters for the experiment adjacent loading at Berth 1 and 3 using 12 rgv’s and unloading at Berth 2.

The basic parameters used for this experiment are shown in Table 24.1. The parameters are the same as used in experiment 10, except for the number of rgv’s used when loading a vessel from an adjacent berth. In this experiment that is the case for Berth 1 and Berth 3.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>fromtob[1,2]</td>
<td>0</td>
</tr>
<tr>
<td>fromtob[3,2]</td>
<td>0</td>
</tr>
<tr>
<td>fromtob[2,1]</td>
<td>1600</td>
</tr>
<tr>
<td>fromtob[2,3]</td>
<td>1600</td>
</tr>
<tr>
<td>inplacecounter 1</td>
<td>0</td>
</tr>
<tr>
<td>inplacecounter 2</td>
<td>0</td>
</tr>
<tr>
<td>inplacecounter 3</td>
<td>0</td>
</tr>
<tr>
<td>seed of uniform</td>
<td>67</td>
</tr>
<tr>
<td>seed of arrivaltimes feeders</td>
<td>17</td>
</tr>
<tr>
<td>seed of arrivaltimes big</td>
<td>27</td>
</tr>
<tr>
<td>import generator feeders</td>
<td>40</td>
</tr>
<tr>
<td>export generator feeders</td>
<td>1050</td>
</tr>
<tr>
<td>import generator big vessels</td>
<td>3150</td>
</tr>
<tr>
<td>export generator big vessels</td>
<td>0</td>
</tr>
<tr>
<td>rgv full speed</td>
<td>3</td>
</tr>
<tr>
<td>rgv low speed passing a shortcut</td>
<td>2</td>
</tr>
<tr>
<td>rgv node speed passing a node</td>
<td>1</td>
</tr>
<tr>
<td>rgv acceleration</td>
<td>1</td>
</tr>
<tr>
<td>minimum distance between trolleys</td>
<td>3</td>
</tr>
<tr>
<td>number of stacking cranes at terminal</td>
<td>70</td>
</tr>
<tr>
<td>number of agv’s per portainer used:</td>
<td></td>
</tr>
<tr>
<td>- unloading a vessel</td>
<td>6</td>
</tr>
<tr>
<td>- loading a vessel from different berth</td>
<td>12</td>
</tr>
<tr>
<td>- shuffling at the terminal</td>
<td>10</td>
</tr>
<tr>
<td>stacking crane cycle time</td>
<td>80</td>
</tr>
<tr>
<td>mean portainer cycle time</td>
<td>58</td>
</tr>
</tbody>
</table>

Table 24.1 input parameters for experiment; Adjacent loading at Berth 1 and Berth 3 using 12 rgv’s and unloading at Berth 2.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>fromtob[1,2]</td>
<td>40</td>
</tr>
<tr>
<td>fromtob[3,2]</td>
<td>40</td>
</tr>
<tr>
<td>fromtob[2,1]</td>
<td>2083</td>
</tr>
<tr>
<td>fromtob[2,3]</td>
<td>2087</td>
</tr>
</tbody>
</table>

Table 24.2 output parameter values for experiment; Adjacent loading at Berth 1 and Berth 3 using 12 rgv’s and unloading at Berth 2.
24.1.1 Adjacent loading on Berths 1& 3 with 12 rgv’s per portainer; unloading on Berth 2

As both feeder vessels will be provided from Yard 2 this means that 1050 * 2 = 2100 containers have to retrieved from these stacks. Both feeder vessels will leave the terminal after 4.3 hours. In this time span the mega vessel has unloaded 1730 containers into Yard 2. Adding these up to the 2100 containers loaded onto both feeder vessels will amount to 3830 containers.

Considering that the number of containers that will retrieved from the 20' stacks and the 40' stacks are the same means that 1915 containers will be retrieved from both stack types.

- Required 20' warehouse crane operational time: 1915 * 80 seconds = 153,200 seconds  
- Required 40' warehouse crane operational time: 1915 * 80 seconds = 153,200 seconds

Number of warehouse cranes in Yard 2:
- Number of warehouse cranes in the 20' stacks in Yard 2 = 6 * 2 = 12 warehouse cranes  
- Number of warehouse cranes in the 40' stacks in Yard 2 = 6 * 3 = 18 warehouse cranes

Needed operational time per warehouse crane:

- \[ \frac{153,200}{12} = 12,767 \text{ seconds per 20'} \text{ stack warehouse crane} \]
- \[ \frac{153,200}{18} = 8,511 \text{ seconds per 40'} \text{ stack warehouse crane} \]

Required adjacent loading time for both feeder vessels: 4.7 hours ⇒ 16,920 seconds

Occupation rates of the warehouse cranes:

20' stack: \[ \left( \frac{12,767}{16,920} \right) \times 100\% = 75.5\% \]

40' stack: \[ \left( \frac{8,511}{16,920} \right) \times 100\% = 50.3\% \]
Appendix 25

Figure 24.1, rgv row at portainer 1

Figure 24.2, rgv row at portainer 2

Figure 24.3, rgv row at portainer 3

Figure 24.4, rgv row at portainer 4
Figure 24.13, rgv row at portainer 13

Figure 24.14, rgv row at portainer 14

Figure 24.15, rgv stack row 101

Figure 24.16, rgv stack row 102
Appendix 25

Figure 24.17, rgv stack row 103

Figure 24.18, rgv stack row 104

Figure 24.19, rgv stack row 105

Figure 24.20, rgv stack row 106
Appendix 25

Figure 24.28, rgv node row 320

Figure 24.29, rgv node row 350

Figure 24.30, rgv node row 504

Figure 24.31, rgv node row 506
Appendix 25: Input parameters for the experiment with disturbances in the portainer cycle times.

The basic parameters used for this experiment are shown in Table 25.1. The parameters are the same as used in experiment 10, except for the portainer cycle times. The portainer cycle times used at the reference terminal are the shortest cycle times possible. In reality these cycle times will not be as perfect. Because that would mean the portainer driver does not make any mistakes which would decrease the cycle times of the portainer. Therefore extra time is added to model disturbances. Two kinds of disturbances are added. A small disturbance with a maximum of 5 seconds and a big disturbance that only occurs once in a hundred moves.

Including these two disturbances, the new mean portainer cycle time is 63 seconds.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>fromtob[1,2]</td>
<td>0</td>
</tr>
<tr>
<td>fromtob[3,2]</td>
<td>0</td>
</tr>
<tr>
<td>fromtob[2,1]</td>
<td>1600</td>
</tr>
<tr>
<td>fromtob[2,3]</td>
<td>1600</td>
</tr>
<tr>
<td>inplacecounter 1</td>
<td>0</td>
</tr>
<tr>
<td>inplacecounter 2</td>
<td>0</td>
</tr>
<tr>
<td>inplacecounter 3</td>
<td>0</td>
</tr>
<tr>
<td>seed of uniform</td>
<td>67</td>
</tr>
<tr>
<td>seed of arrivaltimes feeders</td>
<td>17</td>
</tr>
<tr>
<td>seed of arrivaltimes big</td>
<td>27</td>
</tr>
<tr>
<td>import generator feeders</td>
<td>40</td>
</tr>
<tr>
<td>export generator feeders</td>
<td>1050</td>
</tr>
<tr>
<td>import generator big vessels</td>
<td>3150</td>
</tr>
<tr>
<td>export generator big vessels</td>
<td>0</td>
</tr>
<tr>
<td>rgy full speed</td>
<td>3</td>
</tr>
<tr>
<td>rgy low speed passing a shortcut</td>
<td>2</td>
</tr>
<tr>
<td>rgy node speed passing a node</td>
<td>1</td>
</tr>
<tr>
<td>rgy acceleration</td>
<td>1</td>
</tr>
<tr>
<td>minimum distance between trolleys</td>
<td>3</td>
</tr>
<tr>
<td>number of stacking cranes at terminal</td>
<td>90</td>
</tr>
<tr>
<td>number of agvs per portainer used:</td>
<td></td>
</tr>
<tr>
<td>- unloading a vessel</td>
<td>6</td>
</tr>
<tr>
<td>- loading a vessel from different berth</td>
<td>10</td>
</tr>
<tr>
<td>- shuffling at the terminal</td>
<td>10</td>
</tr>
<tr>
<td>stacking crane cycle time</td>
<td>80</td>
</tr>
<tr>
<td>mean portainer cycle time</td>
<td>63</td>
</tr>
</tbody>
</table>

Table 25.1 input parameters for experiment; Adjacent loading at Berth 1 and Berth 3 and unloading at Berth 2, using variable portainer cycle times.
Figure 25.1, queue under portainer 1 while loading a feeder from an adjacent berth

Figure 25.2, queue under portainer 2 while loading a feeder from an adjacent berth

Figure 25.3, queue under portainer 3 while loading a feeder from an adjacent berth

Figure 25.4, queue under portainer 4 while loading a feeder from an adjacent berth
Appendix 25

Portainer 5 queue

Figure 25.5, queue under portainer 5 while unloading a mega vessel

Portainer 6 queue

Figure 25.6, queue under portainer 6 while unloading a mega vessel

Portainer 7 queue

Figure 25.7, queue under portainer 7 while unloading a mega vessel

Portainer 8 queue

Figure 25.8, queue under portainer 8 while unloading a mega vessel

147
Appendix 25

Figure 25.9, queue under portainer 9 while unloading a mega vessel

Figure 25.10, queue under portainer 10 while unloading a mega vessel

Figure 25.11, queue under portainer 11 while loading a feeder from the adjacent berth

Figure 25.12, queue under portainer 12 while loading a feeder from the adjacent berth
Appendix 25

Figure 25.13, queue under portainer 13 while loading a feeder from the adjacent berth

Figure 25.14, queue under portainer 14 while loading a feeder from the adjacent berth
Appendix 26: Input parameters for the experiment with variable crane cycle times.

The basic parameters used for this experiment are shown in Table 26.1. The parameters are the same as used in experiment 10, except for the crane cycle times. The cycle times in this experiment can differ from 44 up to 116 seconds, with a mean cycle time of 80 seconds.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>fromtob[1,2]</td>
<td>0</td>
<td>containers</td>
</tr>
<tr>
<td>fromtob[3,2]</td>
<td>0</td>
<td>containers</td>
</tr>
<tr>
<td>fromtob[2,1]</td>
<td>1600</td>
<td>containers</td>
</tr>
<tr>
<td>fromtob[2,3]</td>
<td>1600</td>
<td>containers</td>
</tr>
<tr>
<td>inplacecounter 1</td>
<td>0</td>
<td>containers</td>
</tr>
<tr>
<td>inplacecounter 2</td>
<td>0</td>
<td>containers</td>
</tr>
<tr>
<td>inplacecounter 3</td>
<td>0</td>
<td>containers</td>
</tr>
<tr>
<td>seed of uniform</td>
<td>67</td>
<td>-</td>
</tr>
<tr>
<td>seed of arrivaltimes feeders</td>
<td>17</td>
<td>-</td>
</tr>
<tr>
<td>seed of arrivaltimes big</td>
<td>27</td>
<td>-</td>
</tr>
<tr>
<td>import generator feeders</td>
<td>40</td>
<td>containers</td>
</tr>
<tr>
<td>export generator feeders</td>
<td>1050</td>
<td>containers</td>
</tr>
<tr>
<td>import generator big vessels</td>
<td>3150</td>
<td>containers</td>
</tr>
<tr>
<td>export generator big vessels</td>
<td>0</td>
<td>containers</td>
</tr>
<tr>
<td>rgy full speed</td>
<td>3</td>
<td>m/s</td>
</tr>
<tr>
<td>rgy low speed passing a shortcut</td>
<td>2</td>
<td>m/s</td>
</tr>
<tr>
<td>rgy node speed passing a node</td>
<td>1</td>
<td>m/s</td>
</tr>
<tr>
<td>rgy acceleration</td>
<td>1</td>
<td>m/s²</td>
</tr>
<tr>
<td>minimum distance between trolleys</td>
<td>3</td>
<td>m</td>
</tr>
<tr>
<td>number of stacking cranes at terminal</td>
<td>90</td>
<td>-</td>
</tr>
<tr>
<td>number of agv's per portainer used:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>- unloading a vessel</td>
<td>6</td>
<td>-</td>
</tr>
<tr>
<td>- loading a vessel from different berth</td>
<td>10</td>
<td>-</td>
</tr>
<tr>
<td>- shuffling at the terminal</td>
<td>10</td>
<td>-</td>
</tr>
<tr>
<td>mean stacking crane cycle time</td>
<td>80</td>
<td>seconds</td>
</tr>
<tr>
<td>mean portainer cycle time</td>
<td>58</td>
<td>seconds</td>
</tr>
</tbody>
</table>

*Table 26.1 input parameters for experiment; Adjacent loading at Berth 1 and Berth 3 and unloading at Berth 2, using variable crane cycle times.*
Figure 26.1, rgv queue on warehouse lane 71

Figure 26.2, rgv queue on warehouse lane 72

Figure 26.3, rgv queue on warehouse lane 73

Figure 26.4, rgv queue on warehouse lane 74
Figure 26.9, rgv queue on warehouse lane 80
### Appendix 27 Table section

#### AVERAGE WAITING TIME OF CUSTOMERS IN THE QUEUE M/D/1 (IN UNITS OF AVERAGE WAITING TIME)

<table>
<thead>
<tr>
<th>utilization (u)</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>0.0062</td>
<td>0.0009</td>
<td>0.0002</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>0.2</td>
<td>0.0242</td>
<td>0.0066</td>
<td>0.0021</td>
<td>0.0007</td>
<td>0.0002</td>
<td>0.0001</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>0.3</td>
<td>0.0553</td>
<td>0.0201</td>
<td>0.0083</td>
<td>0.0039</td>
<td>0.0019</td>
<td>0.0009</td>
<td>0.0005</td>
<td>0.0002</td>
<td>0.0001</td>
</tr>
<tr>
<td>0.4</td>
<td>0.1033</td>
<td>0.0450</td>
<td>0.0227</td>
<td>0.0124</td>
<td>0.0072</td>
<td>0.0043</td>
<td>0.0026</td>
<td>0.0017</td>
<td>0.0011</td>
</tr>
<tr>
<td>0.5</td>
<td>0.1767</td>
<td>0.0872</td>
<td>0.0497</td>
<td>0.0307</td>
<td>0.0199</td>
<td>0.0135</td>
<td>0.0093</td>
<td>0.0066</td>
<td>0.0047</td>
</tr>
<tr>
<td>0.6</td>
<td>0.2930</td>
<td>0.1584</td>
<td>0.0984</td>
<td>0.0661</td>
<td>0.0467</td>
<td>0.0342</td>
<td>0.0257</td>
<td>0.0197</td>
<td>0.0154</td>
</tr>
<tr>
<td>0.7</td>
<td>0.4936</td>
<td>0.2862</td>
<td>0.1897</td>
<td>0.1355</td>
<td>0.1016</td>
<td>0.0788</td>
<td>0.0627</td>
<td>0.0508</td>
<td>0.0419</td>
</tr>
<tr>
<td>0.8</td>
<td>0.9030</td>
<td>0.5537</td>
<td>0.3860</td>
<td>0.2890</td>
<td>0.2265</td>
<td>0.1833</td>
<td>0.1519</td>
<td>0.1282</td>
<td>0.1098</td>
</tr>
<tr>
<td>0.9</td>
<td>2.0138</td>
<td>1.2887</td>
<td>0.9340</td>
<td>0.7227</td>
<td>0.5848</td>
<td>0.4894</td>
<td>0.4164</td>
<td>0.3606</td>
<td>0.3175</td>
</tr>
</tbody>
</table>

**Table I**

#### AVERAGE WAITING TIME OF CUSTOMERS IN THE QUEUE D/M/1 (IN UNITS OF AVERAGE SERVICE TIME)

<table>
<thead>
<tr>
<th>utilization (u)</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>0.2</td>
<td>0.0003</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>0.3</td>
<td>0.0048</td>
<td>0.0008</td>
<td>0.0002</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>0.4</td>
<td>0.0223</td>
<td>0.0060</td>
<td>0.0019</td>
<td>0.0007</td>
<td>0.0002</td>
<td>0.0001</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>0.5</td>
<td>0.0649</td>
<td>0.0239</td>
<td>0.0103</td>
<td>0.0049</td>
<td>0.0024</td>
<td>0.0013</td>
<td>0.0007</td>
<td>0.0004</td>
<td>0.0002</td>
</tr>
<tr>
<td>0.6</td>
<td>0.1520</td>
<td>0.0685</td>
<td>0.0360</td>
<td>0.0206</td>
<td>0.0125</td>
<td>0.0079</td>
<td>0.0051</td>
<td>0.0034</td>
<td>0.0023</td>
</tr>
<tr>
<td>0.7</td>
<td>0.3257</td>
<td>0.1696</td>
<td>0.1020</td>
<td>0.0665</td>
<td>0.0458</td>
<td>0.0327</td>
<td>0.0240</td>
<td>0.0180</td>
<td>0.0137</td>
</tr>
<tr>
<td>0.8</td>
<td>0.7111</td>
<td>0.4114</td>
<td>0.2725</td>
<td>0.1947</td>
<td>0.1461</td>
<td>0.1134</td>
<td>0.0903</td>
<td>0.0734</td>
<td>0.0605</td>
</tr>
<tr>
<td>0.9</td>
<td>1.9330</td>
<td>1.2112</td>
<td>0.8612</td>
<td>0.6567</td>
<td>0.5238</td>
<td>0.4310</td>
<td>0.3629</td>
<td>0.3110</td>
<td>0.2703</td>
</tr>
</tbody>
</table>

**Table II**
<table>
<thead>
<tr>
<th>( \lambda ) = ( 1/\mu )</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>0.5796</td>
<td>0.2032</td>
<td>0.1643</td>
<td>0.1200</td>
<td>0.0951</td>
<td>0.0793</td>
<td>0.0684</td>
<td>0.0586</td>
<td>0.0547</td>
<td>0.0501</td>
</tr>
<tr>
<td>0.2</td>
<td>0.6786</td>
<td>0.2488</td>
<td>0.2163</td>
<td>0.1875</td>
<td>0.1589</td>
<td>0.1354</td>
<td>0.1175</td>
<td>0.1020</td>
<td>0.0907</td>
<td>0.0814</td>
</tr>
<tr>
<td>0.3</td>
<td>0.7996</td>
<td>0.2627</td>
<td>0.2330</td>
<td>0.2088</td>
<td>0.1800</td>
<td>0.1529</td>
<td>0.1306</td>
<td>0.1116</td>
<td>0.1008</td>
<td>0.0903</td>
</tr>
<tr>
<td>0.4</td>
<td>0.9213</td>
<td>0.2748</td>
<td>0.2452</td>
<td>0.2234</td>
<td>0.1970</td>
<td>0.1726</td>
<td>0.1494</td>
<td>0.1280</td>
<td>0.1159</td>
<td>0.1053</td>
</tr>
<tr>
<td>0.5</td>
<td>0.9635</td>
<td>0.2826</td>
<td>0.2517</td>
<td>0.2330</td>
<td>0.2079</td>
<td>0.1846</td>
<td>0.1620</td>
<td>0.1410</td>
<td>0.1271</td>
<td>0.1159</td>
</tr>
<tr>
<td>0.6</td>
<td>1.0062</td>
<td>0.2918</td>
<td>0.2602</td>
<td>0.2449</td>
<td>0.2209</td>
<td>0.1983</td>
<td>0.1767</td>
<td>0.1562</td>
<td>0.1435</td>
<td>0.1320</td>
</tr>
<tr>
<td>0.7</td>
<td>1.1492</td>
<td>0.3019</td>
<td>0.2761</td>
<td>0.2627</td>
<td>0.2419</td>
<td>0.2222</td>
<td>0.2034</td>
<td>0.1846</td>
<td>0.1716</td>
<td>0.1590</td>
</tr>
<tr>
<td>0.8</td>
<td>1.2936</td>
<td>0.3124</td>
<td>0.2931</td>
<td>0.2824</td>
<td>0.2646</td>
<td>0.2481</td>
<td>0.2322</td>
<td>0.2164</td>
<td>0.1986</td>
<td>0.1853</td>
</tr>
<tr>
<td>0.9</td>
<td>1.3982</td>
<td>0.3234</td>
<td>0.3002</td>
<td>0.2925</td>
<td>0.2774</td>
<td>0.2625</td>
<td>0.2484</td>
<td>0.2335</td>
<td>0.2179</td>
<td>0.2042</td>
</tr>
<tr>
<td>1.0</td>
<td>1.5000</td>
<td>0.3350</td>
<td>1.0000</td>
<td>0.3395</td>
<td>0.3025</td>
<td>0.2876</td>
<td>0.2737</td>
<td>0.2607</td>
<td>0.2483</td>
<td>0.2355</td>
</tr>
</tbody>
</table>

Table III
Study on the Designing and Simulation of a High capacity transshipment terminal

- Appendix 28 Listing

Impression of the portainers and container warehouse

December 2000
Hein van Hees
Thies Visscher

TU Delft
Delft University of Technology
Faculty of Civil Engineering and Geosciences
Department of Hydraulic and Geotechnical Engineering
Section of Hydraulic Engineering
28-I. Define section
28-II. Modules
28-III. Macro's
28-IV. Clarifying word list
28-V. Input file for lay out of the terminal
28-VI Input file Rgv attributes
DEFINE
  COMPONENT
    MAIN

  INTEGER
    N
    I
    SHUFFLE_COUNTER
    A
    BERTH_NR
    FROM_TO( 3, 3)
    Q
    J
    N_NODES
    SUPER_NODE
    PTL
    INPLACE_COUNTER( 3)

  LOGICAL
    QUAY_LOCATION( 18)

  REAL
    K
    SHIP_TURN_AROUND_TIME
    N_IN
    N_OUT
    ACC
    FULL_SPEED
    LOW_SPEED
    NODE_SPEED
    HANDLING
    DISTANCE
    MN
    VNRM
    MNBV
    NM
    MNB
    BMN
    C1
    C2
    C3
    C4
    C5
    C6
    C7
    C8
    C9
    C10

  CHARACTER
    SHIP_TIME( 12)
    PLACES( 4)
    OUT_STRING( 14)
    CHAR( 22)
    CHAR( 9)

  REFERENCE TO DATASTREAM
    OUT_PORT
    OUT_PORT_U
    HCONFIG
    OUT_PORTATION( 300)

  REFERENCE TO POINTSTREAM
    SDWELL_TIMES
    AVG_CYCLE_TIMES
    OUT
    SHIP_TURN_AROUND_TIMES
    DOORLOOP

  REFERENCE TO DISTRIBUTION
ARRIVALTIMES
UNIF
ANIF
ARRIVALTIMESBIG
REFERENCE TO WINDOW
AGVCYCLTIMES
SHIPDWELLTIMES
SHTURNAROUNDTIMES
CHART
DOORWIN
ANCHORAGETIMES
PORTAINER1QUEUE
BERTHOCUPIED{ 3]
PORTAINERSACTIVE
PORT18QUEUE
PORT17QUEUE
PORT16QUEUE
PORT15QUEUE
PORT14QUEUE
PORT13QUEUE
PORT12QUEUE
PORT11QUEUE
PORT10QUEUE
PORT9QUEUE
PORT8QUEUE
PORT7QUEUE
PORT6QUEUE
PORT5QUEUE
PORT4QUEUE
PORT3QUEUE
PORT2QUEUE
PORT1QUEUE
CONTROL
CRANE2QUEUE
SECROW[200]
REFERENCE TO WOB
CONS
B_OK
FRM_MT
FRM_M
FRM_ST
FRM_MN
STR_MT
STR_M
STR_MP
STR_MN
FRM_MAXN
STR_TEMP
STR_MAXN
FRM_MP
FRM_TEMP
STR_01
STR_02
STR_03
STR_04
STR_05
STR_06
STR_07
STR_08
STR_09
STR_10
FRM_10
FRM_01
REFERENCE TO SET
FROMPARKLIST
CLAIMSET
ANCHORAGE
QUAYRAIL
BERTH[ 3]
AGVPARKING
CRANELIST
AGVWORKLIST
ACTIVEAGVLIST
LOADINGSHIP
UNLOADINGSHIP
PORTLOCATIONS
ACTIVESHIPLIST
QUAYWORKLIST
BERTHAGWAITLIST
PORTAINERACTIVELIST
PLATFORMS
SECTIONS
STACKLIST
NODE_LIST
AGVBERTHSHUFFLELIST[ 3]
PORTAINERPERBERTH[ 3]
GONESHIPLIST
REFERENCE TO PORTAINER
NEWPORTAINER
REFERENCE TO AGV
NEWAGV
REFERENCE TO CRANE
NEWCRANE
REFERENCE TO PORTLOC
NEWPORTLOC
REFERENCE TO STACK
NEWSTACK
STACKS[ 40]
REFERENCE TO NODE
NODES[ 800]
REFERENCE TO SECTION
NEWSECTION

SHIPGEN
INTEGER
S
R
REAL
TIMEBIG
TIMESMALL
REFERENCE TO SHIP
NEWSHIP
BERTHAMASTER
LOGICAL
OCCUPIED[ 3]
REFERENCE TO SHIP
BERTHAMASTERSHIP
PORTAINERMASTER
INTEGER
E
ACTIVEPORT
REAL
TIME
REFERENCE TO SHIP
PTMSHIP
REFERENCE TO PORTAINER
PORTAINERMASTERPORTAINER
REFERENCE TO PORTLOC
PMPORTLOC
IMPORTGENERATOR
REAL
WW
REFERENCE TO SHIP
IMPORTSHIP
EXPORTGENERATOR
REFERENCE TO SHIP
EXPORTSHIP
AGVMASTER
INTEGER
PAGV
PASSAGV
ACTAGV
ACTIVEAGV
ASL
NEEDEDAGVS
PAL
RI
RA
RO
RU
REFERENCE TO SHIP
AMSHIP
REFERENCE TO PORTAINER
AGVPORTAINER
REFERENCE TO AGV
BYEAGV
HIAGV
DEBYEAGV
TRAFFICMASTER
INTEGER
V4
V3
V2
V1
WHERE
TMX
TMSHIPNUMBER
TMTOTAL
TMTOTALIM
TMTOTALEX
TMY
TMYY
TMXX
REAL
LASTK
CYCLETIME
TMRES40IM
TMRES40EX
W
TMRESTOTEX
W1
W2
W3
TMRESTOTIM
W5
W4
W6
REFERENCE TO SHIP
TMSHIP
REFERENCE TO PORTAINER
TMPORT
REFERENCE TO AGV
TMAGV
REFERENCE TO CRANE
TMCRANE
REFERENCE TO PORTLOC
TMPORTLOC
REFERENCE TO STACK
GOEDE_STACK
REFERENCE TO NODE
GOEDE_NODE
REFERENCE TO SECTION
GOEDE_SECTION

CLASS
SHIP
INTEGER
FIRSTSHUFFLEAGV
SHIPBERTHNUMBER
SHIPNUMBER
TOTALLOAD
TOTALUNLOAD
IM20
EX40
IM40
EX20
TOTAL
TEX
TOTALU
REAL
SHIPDWEELL
SHIPANCHORAGETIME
SHIPARRIVALTIME
PROPEX20
PROPIM40
PROPEX40
PROPIM20
PORTAINER
INTEGER
PORTAINERNUMBER
C
PTBERTHNR
T
H
PORTAINERSHIPNB
FUTPORTAINERSHIPNB
ACTLOCNR actual location number
FUTLOCNR future location number
L
PP
AGVPPNEEDED
REP
PERS
PER
TT
REAL
QQ
REFERENCE TO SET
PORTQUEUE
PORTASQUEUE
AGVPPACTIVE_ODD
AGVPPACTIVE
AGVPPACTIVER
REFERENCE TO SHIP
PTSHIP
REFERENCE TO AGV
PORTBYEAGV
ASAGV
PORTAGV
REFERENCE TO PORTLOC
PTPORTLOC

AGV

INTEGER
NR
SBNR
AGVNUMBER
XX
Y
AGVLOC
AGVBERTHNB
YY
LE
X
LL
YYY
XXX
ZZZ
RR
BB

LOGICAL
CONT20SHUF
CONT20FT
WEIRD
EASYLOAD
LOADEDAGV
YARD
BYE
FIRSTSHUFFLE
DELAYED
Z
AGVBIG
AGVRED
AGVGREEN
AGVBLUE
SHUFFLEAGV
UNEASY
BEENINSTACK

REAL
FREE_WAY
CLAIMEDLENGTH
REACHABLE_POINT
B
ARRIVTIME
REFERENCE TO SET
ROUTE
ENTRANCE
SUPERROUTE
REFERENCE TO AGV
HEADER
REFERENCE TO CRANE
AGVCRANE
REFERENCE TO PORTLOC
AGVPORLLOC
REFERENCE TO STACK
AGVSTACK
REFERENCE TO NODE
EXIT
SUPEREXIT
ENTRANCE_NODE
BEFORE_ENTRANCE_NODE
REFERENCE_NODE
CLAIM_NODE
CLAIM2
CLAIM1
LASTNODE
EXITNODE
CUR_NODE
REFERENCE TO SECTION
CLAIM_SEC
CUR_SEC
AGV_SECTION
REFERENCE TO PLATFORM
STOP
CRANE
INTEGER
CRANECLUSTERNR
COUNTER
CRANENUMBER
AANTALAGV
READYCNT
KLAARCOUNT
LOGICAL
SLEEP
REAL
CONTAINERS
F
G
CHARACTER
CRANENAME[8]
REFERENCE TO SET
CRANEQUEUE
READYQUEUE
CRANECOSSENQUEUE
CRANELADENQUEUE
ASSIGNEDQUEUE
REFERENCE TO AGV
CRANEAGV
ASSIGNEDAGV
REFERENCE TO STACK
CRANESTACK
PORTLOC
INTEGER
PORTLOCNUMBER
TELLER401M
TELLER201M
REFERENCE TO SET
PORTLOQUEUE
LOCATIONSET
ASSIGNEDAGVS
STACK
INTEGER
STACKNUMBER
P
D
REAL
   ST_X
   ST_Y
   STACKCOUNT
   BGCOUNT
   SGCOUNT
   BRCOUNT
   SRCOUNT
   BSCOUNT
   SBCOUNT
REFERENCE TO SET
   WANTASQUEUE
   STACK_ROW
REFERENCE TO AGV
   STACKAGV
REFERENCE TO CRANE
   THECRANE
REFERENCE TO NODE
   IN_STACK_NODE
REFERENCE TO SECTION
   WAY_IN_STACK
   WAY_OUT_STACK

NODE
INTEGER
   CLAIMED
   NODEAGVNB
   NODE_NUMBER
REAL
   ND_X
   ND_Y
   CLAIMTIME
   EXITTIME
   NODESPEED
   NODEDISTANCE1
   NODEDISTANCE2
MACRO
   SIDE_TRACK[ 2]
REFERENCE TO AGV
   NODEAGV
REFERENCE TO SECTION
   OUTS[ 3]
SECTION
REAL
   S_LENGTH
   SEC_ROW_NUMBER
REFERENCE TO SET
   SEC_ROW
REFERENCE
   END_POINT
PLATFORM
REAL
   PL_X
   PL_Y
REFERENCE TO POINTSTREAM
   INTERVAL_CHECK
REFERENCE TO SET
   ROW
REFERENCE TO AGV
   AGVPLATFORM
REFERENCE TO NODE
   IN_NODE
REFERENCE TO SECTION
WAY_IN
WAY_OUT
EXTERNAL
TIMEUNIT
SECOND
### Appendix 28-II Modules

<table>
<thead>
<tr>
<th>Mod</th>
<th>MODULES</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MAINMOD</td>
</tr>
<tr>
<td></td>
<td>GENMOD</td>
</tr>
<tr>
<td></td>
<td>BERTHMASTMOD</td>
</tr>
<tr>
<td></td>
<td>SHIPMOD</td>
</tr>
<tr>
<td></td>
<td>PORTAINERMASTMOD</td>
</tr>
<tr>
<td></td>
<td>AGVMASTMOD</td>
</tr>
<tr>
<td></td>
<td>PORTAINERMOD</td>
</tr>
<tr>
<td></td>
<td>EXPORTGEN</td>
</tr>
<tr>
<td></td>
<td>IMPORTGEN</td>
</tr>
<tr>
<td></td>
<td>AGVMOD</td>
</tr>
<tr>
<td></td>
<td>TRAFFICMASTMOD</td>
</tr>
<tr>
<td></td>
<td>CRANEMOD</td>
</tr>
<tr>
<td></td>
<td>NODEMOD</td>
</tr>
</tbody>
</table>
The main starts with the creating of different sets.

Node_list is a set containing all the nodes

Agvberthshufflelist [1 to 3] are 3 sets containing all the rgv's shuffling from that berth

Activeshiplist a set containing the active vessels
Stacklist a set containing all the stacks
Agvparking a set containing the not active rgv's
Agvworklist a set containing rgv's with work requests for the trafficmaster

Cranelist a set containing all the cranes
Loadingship a set containing all the loading vessels at the terminal

Outportation creates the output files
Goneshiplist a set containing the terminated vessels, in this way their attributes can be stored

Portainerperberth [1 to 3] keeps count of the active portainers per berth

Portaineractivelist a set containing all the active portainers
Portlocations a set containing all the portainerlocations

Quayrail a set containing all the portainers

Occupied, is a logical if a berth is taken by a vessel occupied becomes true

Unloadingship a set containing all the vessels that are unloading

In line 57 to 72 the portainers are created

n = number of portainers that have to be created
1 @set creation@  
2 r-0  
3 doorloop ← NEW POINTSTREAM  
4 node_list ← NEW SET  
5 activeagvlist ← NEW SET WITH HISTORY  
6 agvberthshuflelist[1] ← NEW SET  
7 agvberthshuflelist[2] ← NEW SET  
8 agvberthshuflelist[3] ← NEW SET  
9 activeshiplist ← NEW SET  
10 stacklist ← NEW SET  
11 agvparking ← NEW SET WITH HISTORY  
12 agworklist ← NEW SET  
13 claimset ← NEW SET  
14 agvcycletimes ← NEW POINTSTREAM  
15 anchorage ← NEW SET WITH HISTORY  
16 berth[1] ← NEW SET WITH HISTORY  
17 berth[2] ← NEW SET WITH HISTORY  
18 berth[3] ← NEW SET WITH HISTORY  
19 berthagvwaitlist ← NEW SET  
20 cranelist ← NEW SET  
21 loadingship ← NEW SET  
22 outport ← NEW DATASTREAM  
23 outportu ← NEW DATASTREAM  
24 geluk ← NEW POINTSTREAM  
25 out ← NEW POINTSTREAM  
26 JUMP TO create_outportation  
27 goneshiplist←NEW SET  
28 ATTACH "shiparrivaltimes.txt" TO outportu  
29 ATTACH "concout.txt" TO outport  
30 portainerperberth[3] ← NEW SET WITH HISTORY  
31 portainerperberth[2] ← NEW SET WITH HISTORY  
32 portainerperberth[1] ← NEW SET WITH HISTORY  
33 portaineractivelist ← NEW SET WITH HISTORY  
34 portlocations ← NEW SET  
35  
36 geluk ← NEW POINTSTREAM  
37 out ← NEW POINTSTREAM  
38  
39  
40 quayrail ← NEW SET  
41 quayworklist ← NEW SET  
42  
44 occupied[1] ← FALSE  
47  
48 unloadingship ← NEW SET  
49  
50 sdwelltimes ← NEW POINTSTREAM  
51 shipturnaroundtimes ← NEW POINTSTREAM  
52  
53 k ← 0.225  
54 s ← 1  
55  
56 @creating the portainers:@
57 @creating the portainers:@
58  
59 n ← 18010  
60  
62 FOR i ← 1 TO n
actlocnr is the actual location number, the location where the portainer is at that moment

agvpactive_odd is a set containing the number of odd agv's per portainer active

agvpactive is a set containing the number of agv's (rgv's) per portainer active

From line 74 to 95 portainer locations are created

n is the number of portainerlocations to be created

portloc is short for portainerlocation

quaylocation[i] is true means this location is taken by a portainer

From line 96 to 112 the rgv's and their attributes are created

n is the number of rgv's that have to be created

Route of the rgv is a set containing all the exits of the agv

Entrance of the rgv is a set containing the starting point of a route of the rgv

x refers to the current portlocation of the rgv

y refers to the current stack of the rgv

Loadedagv is a logical, false means the rgv is not carrying a container

From line 115 to 150 the cranes are created

n is the number of cranes that have to be created
newportainer = NEW portainer
portainerNumber OF newportainer = i
actioacr OF newportainer = i
portqueue OF newportainer = NEW SET WITH HISTORY
portasqueue OF newportainer = NEW SET
agvppactive_odd OF newportainer = NEW SET
agvppactive OF newportainer = NEW SET
agvppactivev OF newportainer = NEW SET
JOIN newportainer TO quayrail
END

n = 18
FOR i = 1 TO n
newportloc = NEW portloc
portlocnumber OF newportloc = i
portlocqueue OF newportloc = NEW SET WITH HISTORY
assignedags OF newportloc = NEW SET
JOIN newportloc TO portlocations
IF i <= 10
  quaylocation[i] = TRUE
END
IF i > 10
  quaylocation[i] = FALSE
END
locationset OF newportloc = NEW SET
teller20im OF newportloc = 0
teller40im OF newportloc = 0
END

n = 2008140
FOR i = 1 TO n
newavg = NEW agv
route OF newavg = NEW SET
superroute OF newavg = NEW SET
entrance OF newavg = NEW SET
z OF newavg = FALSE
x OF newavg = 2
y OF newavg = 42
agvnumber OF newavg = i
loadedagv OF newavg = FALSE
JOIN newavg TO agvparking
END

@ creating the cranes@

n = 180
FOR i = 1 TO n
newcrane = NEW crane
cranagey OF newcrane = FIRST agv IN agvparking WITH agvnumber = 200
cranenumber OF newcrane = i
cranequeue is a set containing the rgv's waiting to be served by the crane
assignedqueue is a set containing rgv's assigned to the crane. When a rgv is assigned
the crane can start working if possible

cranelist a set containing all the cranes

fromtob[1,2] is 4000 tells there are 4000 containers in the stacks behind berth 1
which are destined to be taken away by a vessel from berth 2

inplacecounter[1] counts how many containers are destined to be taken away
from that berth by a vessel arriving at that berth. The containers can be loaded easy
when a vessel arrives at that berth.

The feeders are generated

The mega vessels are created

From line 178 to 199 the input parameters are imported from a file
This file is called hconfig

acc is the acceleration of the rgv's to be used during the run
counter OF newcrane ← 0
aantalag OF newcrane←0
klarcount OF newcrane ← 0
IF ((i-1)MODULO(10))+1 ≤ 4
  cranename OF newcrane ← "40"
cranenumber OF newcrane ← ptt
END
IF ((i-1)MODULO(10))+1 > 4 & ((i-1)MODULO(10))+1 ≤ 9
  cranename OF newcrane ← "20"
cranenumber OF newcrane ← ptt
END
IF ((i-1)MODULO(10))+1 = 10
  cranename OF newcrane ← "20"
cranenumber OF newcrane ← ptt
ptt ← ptt + 1
END

joinqueue OF newcrane ← NEW SET
assignedqueue OF newcrane ← NEW SET
cranelossenqueue OF newcrane ← NEW SET
craneladenqueue OF newcrane← NEW SET
readyqueue OF newcrane ← NEW SET
containers OF newcrane ← 10
JOIN newcrane TO cranelist

ACTIVATE EACH crane IN cranelist FROM begin IN cranemod

shufflecounter←0
fromtb[1,2]←4000
fromtb[2,2]←4000
fromtb[2,1]←4000
fromtb[2,3]←4000
inplacecounter[1]←4000
inplacecounter[2]←0
inplacecounter[3]←0

unif ← NEW DISTRIBUTION
ATTACH UNIFORM(0,1) TO unif
SEED OF unif ← 67

@Feeders@
arrivaltimes ← NEW DISTRIBUTION
ATTACH NORMAL(10,0) TO arrivaltimes
SEED OF arrivaltimes ← 17

@Mega Vessels@
arrivaltimesbig ← NEW DISTRIBUTION
ATTACH NORMAL(15,0) TO arrivaltimesbig
SEED OF arrivaltimes ← 27

@ creating the streetplan @

hconfig ← NEW DATASTREAM
ATTACH READ FROM CASE TO hconfig
OPEN hconfig FOR INPUT
acc ← READ FROM CASE
full_speed is the top speed of the rgy, this speed can be reached on the main tracks
nodespeed is the speed of a rgy when it crosses a node
low_speed is the speed of the rgy in the stacks and at the short cuts

distance is the minimum distance between rgy's when driving on the terminal

From line 200 to 272 the main tracks of the terminal are created

In line 206, 800 nodes are created, these shape the framework of the terminal

nd_x is the x co-ordinate of node 1
nd_y is the y coordinate of node 1

between nodes sections can be created

end_point is a reference to the last node of a section
s_length is the length of section calculated by the function distance to
sections is a set containing all the sections
different section require different driving speeds
Node 1 to 118 and 501 to 618 create main routes, the driving speed at the main route
is full_speed
node_distance refers to the safety distance at the tracks, this can differ per track
187 full_speed ← READ FROM CASE
188 node_speed ← READ FROM CASE
189 low_speed ← READ FROM CASE
190 acc ← acc8*3600
191 full_speed ← full_speed @*60
192 node_speed ← node_speed @*60
193 low_speed ← low_speed @*60
194 handling ← READ FROM CASE
195 distance ← READ FROM CASE
196
197 sections ← NEW SET CALLED "sections"
198 platforms ← NEW SET CALLED "platforms"
199
200 @@@@@@@@@@@@@@@@@@@
201 @ main track @
202 @@@@@@@@@@@@@@@@@@@
203
204 n=800
205
206 FOR i ← 1 TO n
207 nodes[i] ← NEW node
208 node_number OF nodes[i]=i
209 JOIN nodes[i] TO node_list
210 END
211
212 supernode ← 1
213 lator_gast:
214 i ← READ FROM hconfig
215 supernode ← i
216
217 WHILE i > 0
218 THIS node ← nodes[i]
219 @WAIT 1
220 nd_x ← READ FROM hconfig
221 nd_y ← READ FROM hconfig
222 IF i¹20 & i¹60 & i¹81 & i¹100 & i¹501 & i¹520 & i¹541 & i¹561
223 & i¹20 & i¹600 & i¹201 & i¹218 & i¹231 & i¹248 & i¹261 & i¹278 & i¹291
224 & i¹338 & i¹351 & i¹368 & i¹381 & i¹398 & i¹411 & i¹428 & i¹441 & i¹451
225 THIS section ← NEW section
226 outs[i] OF nodes[i-1] ← THIS section
227 end_point ← THIS node
228 s_length ← distance_to(nd_x OF nodes[i-1],
229 nd_y OF nodes[i-1])
230 JOIN THIS section TO sections
231 END
232
233 IF (i ≥ 1 & i ≤ 118) | (i ≥ 501 & i ≤ 618)
234 nodespeed OF THIS node ← full_speed
235 exittime OF THIS node ← 2*(SQRT(((3/2)*2)/acc)) @3m is de breedte van een
236 nodedistance1 OF THIS node ← (full_speed*full_speed)/(2*acc)
237 nodedistance2 OF THIS node ← ((2*exittime OF THIS node)+(2*8)+2*(nodedist.
238 END
239
240 IF (i ≥ 201 & i ≤ 483)
241 nodespeed OF THIS node ← node_speed
242 nodedistance1 OF THIS node ← (node_speed*node_speed)/(2*acc)
243 nodedistance2 OF THIS node ← nodedistance1 OF THIS node
244 exittime OF THIS node ← 2*(SQRT(16/acc)) @16 lengte va:
245 END
246
247 n_nodes ← i
248 i ← READ FROM hconfig
In line 263 to 273 the stacks are created

In line 274 to 283 platforms are created

In line 284 to 303 the short cuts are created (like the main tracks and the stacks and the platforms these consist of sections as well

In line 304 to 321 supershort cuts are created these act the same as shortcuts but are different because of their location at the terminal
249 END
250
251 THIS node ← nodes[ n_nodes]
252 THIS section ← NEW section
253 outs[1] ← THIS section
254 end_point ← nodes[ supernode]
255 s_length ← distance_to(nd_x OF end_point, nd_y OF end_point)
256
257 JOIN THIS section TO sections
258
259 i ← READ FROM hconfig
260
261 GOTO lator_gast IF i>0
262
263 @ stacks @
264 @ platforms @
265 @ short cuts @
266
267 i-1
268 char ← READ FROM hconfig
269 WHILE char ≠ "END"
270 JUMP TO create_stack @ supplier
271 char ← READ FROM hconfig
272 END
273
274 @ platforms @
275 @ short cuts @
276
277 char ← READ FROM hconfig
278 WHILE char ≠ "END"
279 JUMP TO create_platform
280 char ← READ FROM hconfig
281 END
282
283 @ short cuts @
284
285 i ← READ FROM hconfig
286 WHILE i > 0
287
288 THIS NODE ← NODES[ I]
289 @WAIT 1
290 THIS section ← NEW section
291 sec_row ← NEW SET WITH HISTORY
292 sec_row_number ← i
293 outs[2] ← THIS section
294 side_track[1] ← short_cut @ macro
295 i ← READ FROM hconfig
296 end_point ← nodes[ i]
297 s_length ← distance_to(nd_x OF end_point, nd_y OF end_point)
298
299 JOIN THIS section TO sections
300 i ← READ FROM hconfig
301 END
302
303 @ super short cuts @
304
305 i ← READ FROM hconfig
306 WHILE i > 0
307
308 THIS NODE ← NODES[ I]
309
310
create_portout is a macro that creates all the portainer outportations

conversation is a macro that creates the control box

creating_sections is a macro that creates sections and attaches these sections to windows

Terminate cancels all the operations at the terminal
@WAIT 1
THS section ← NEW section
sec_row ← NEW SET
outs[3] ← THIS section
side_track[2] ← super_shortcut
i ← READ FROM hconfig
end_point ← nodes[i]
s_length ← distance_to(nd_x OF end_point, nd_y OF end_point)
JOIN THIS section TO sections
i ← READ FROM hconfig
END

@-----------------------------@
Start simulation by activating the ship generator@
@-----------------------------@

anchoragetimes← NEW WINDOW CALLED "anchorage"
ATTACH HISTORY OF anchorage TO anchoragetimes
berthoccupied[1] ← NEW WINDOW CALLED "berth1occu"
ATTACH HISTORY OF berth[1] TO berthoccupied[1]
berthoccupied[2] ← NEW WINDOW CALLED "berth2occu"
berthoccupied[3] ← NEW WINDOW CALLED "berth3occu"
JUMP TO_create_portout
JUMP TO conversation
JUMP TO creating_sections
portainersactive ← NEW WINDOW CALLED "active portainers"
ATTACH HISTORY OF unloadingship TO portainersactive
shipdwelltimes ← NEW WINDOW CALLED "shipdwelltimes"
ATTACH HISTORY OF loadingship TO shipdwelltimes
ACTIVATE shipgen FROM generate IN genmod
WAIT 30 HOURS

@-----------------------------@
End the simulation by cancelling the ship generator@
@-----------------------------@
CANCEL shipgen
TERMINATE
The generator is responsible for generating the interarrival times of the vessels

Timebig refers to the interarrival time of a mega vessel

Timesmall refers to the interarrival time of a feeder

if line 7 is true this means there will be a feeder generated first
The generator waits the interarrival time before generating the feeder

Big is an attribute of ship of the type logical, "false" means this is a feeder
Anchorage is a set containing the waiting vessels

timebig < timesmall means the mega vessel will be generated first
the generator waits the interarrivaltime before generating a mega vessel
1 generate:
2
3 timebig ← SAMPLE FROM arrivaltimesbig
4 timesmall ← SAMPLE FROM arrivaltimes
5
6 begin:
7 IF timebig ≥ timesmall
8    WAIT timesmall
9    timebig-timebig-timesmall
10   newship←NEW ship
11   shipnumber OF newship ← s
12   r←r+1
13   big OF newship ← FALSE TRUE
14   JOIN newship TO anchorage
15
16   ACTIVATE newship FROM begin IN shipmod
17   s ← s+1
18   timesmall ← SAMPLE FROM arrivaltimes
19   REPEAT FROM begin
20 END
21
22 IF timebig < timesmall
23    WAIT timebig
24    timesmall-timesmall-timebig
25   newship←NEW ship
26   shipnumber OF newship ← s
27   big OF newship ← TRUE
28   JOIN newship TO anchorage
29   ACTIVATE newship FROM begin IN shipmod
30   s ← s+1
31   timebig ← SAMPLE FROM arrivaltimes@9000
32   REPEAT FROM begin
33 END
If the anchorage is not empty implies there is a vessel waiting with a berth request

Occupied[1] is a logical if occupied [1] is true berth1 is occupied by a vessel

If big is false this is a feeder and has to go to berth 1 or 3

The vessel can only go to berth 1 if this berth is not taken

In line 9 to 20 the berthmaster assigns the free berth 1 to the feeder

the berthmaster tags berth 1 as taken

shipanchoragetime is an attribute of the vessel referring to the time in berth

If the vessel in the anchorage is a mega vessel berth 2 will be assigned

If the vessel in the anchorage is a feeder and berth 1 is already taken berth 3 will be assigned to the feeder
BEGIN:

IF anchorage IS NOT EMPTY
  berthmastership ← FIRST ship IN anchorage

IF big OF berthmastership = FALSE

  IF occupied[1] = FALSE
    occupied[1] ← TRUE
    shipberthnumber OF berthmastership ← 1
    JOIN berthmastership TO unloadingship
    JOIN berthmastership TO berth[1]
    shipanchoragetime OF berthmastership ← NOW
    JOIN berthmastership TO activeshiplist
    REMOVE FIRST ship IN anchorage FROM anchorage
    ACTIVATE agvmaster FROM byeshuffle IN agvmsmod IF agvmaster IS NOT TRUE.
    GOTO begin

END

END

IF big OF berthmastership = TRUE


  occupied[2] ← TRUE
  shipberthnumber OF berthmastership ← 2
  JOIN berthmastership TO unloadingship
  JOIN berthmastership TO berth[2]
  shipanchoragetime OF berthmastership ← NOW
  JOIN berthmastership TO activeshiplist
  REMOVE FIRST ship IN anchorage FROM anchorage
  ACTIVATE agvmaster FROM byeshuffle IN agvmsmod IF agvmaster IS NOT TRUE.
  GOTO begin

END

END

IF big OF berthmastership = FALSE & TRUE

beter:

    occupied[3] ← TRUE
    shipberthnumber OF berthmastership ← 3
    JOIN berthmastership TO unloadingship
    JOIN berthmastership TO berth[3]
    shipanchoragetime OF berthmastership ← NOW
    JOIN berthmastership TO activeshiplist
    REMOVE FIRST ship IN anchorage FROM anchorage
    ACTIVATE agvmaster FROM byeshuffle IN agvmsmod IF agvmaster IS NOT TRUE.
    GOTO begin

END

END

quayfull:

END

ACTIVATE berthmastership FROM berthing IN shipmod IF berthmastership IS NOT ACTIV

PASSIVATE
1

2

3

4

5

6

7

8 importgenerator generates the import containers

9

10

11

12 Exportgenerator generates the export containers

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

31 Goneshiplist is a set containing the vessels that have occupied the berths.

32

33

34

35

36

37

38

39
begin:
shiparrivaltime ← ARRIVALTIME OF THIS ship

firstshuffleavg OF THIS ship ← 1

import:
ACTIVATE importgenerator FROM begin IN importgen
WAIT

export:
ACTIVATE exportgenerator FROM begin IN exportgen
WAIT

berthpermission:
ACTIVATE berthmaster FROM begin IN berthmastmod
WAIT

berthing:
@WAIT 1 HOUR @berthing
ACTIVATE portainermaster FROM begin IN portainermastmod
WAIT

leaving:
occupied( shipberthnumber ) ← FALSE
WAIT 1 SECONDS

shipdwell ← NOW - ARRIVALTIME OF THIS ship
shipturnaroundtime ← NOW - shipanchoragetime OF THIS ship

ENTER goneshiplist

REMOVE THIS ship FROM activeshiplist
REMOVE THIS ship FROM loadingship
REMOVE FIRST ship IN berth[ shipberthnumber ] FROM berth[ shipberthnumber ]
ACTIVATE berthmaster FROM begin IN berthmastmod
TERMINATE
The moment the portainermaster is activated he cancels all the portainers. **Line 3 to 8**

**portainermasterportainer** is an attribute of the portainermaster and a reference to portainer

The portainermaster removes all the portainers from the sets portainerperberth and portaineractivelist

In **line 29 to 252** The portainermaster allocates the portainers to their positions which depend on the occupancy of the berths. All the different options are treated. In this scenario 10 portainers are used for 14 positions in the reference terminal 14 portainers are used.

**futlocnr** is the future location number of the portainer. The new location of the portainer.
begin:
FOR a ← 1 TO 18 @10
  portainermasterportainer ← FIRST portainer IN quayrail WITH portainernumber =
  IF portainermasterportainer IS ACTIVE
    CANCEL portainermasterportainer
END
FOR a ← 1 TO 18
  pmportloc ← FIRST portloc IN portlocations
  REMOVE EACH portainer IN locationset OF pmportloc FROM locationset OF pmportloc
  REMOVE pmportloc TO portlocations
  JOIN pmportloc TO portlocations
END
REMOVE EACH portainer IN portaineractivelist FROM portaineractivelist
FOR e ← 1 TO 3
  REMOVE EACH portainer IN portainerper berth[e] FROM portainerper berth[e]
END
  WAIT
END
  IF big OF FIRST OF berth[1] = FALSE
    FOR a ← 3 TO 6
      portainermasterportainer ← FIRST portainer IN quayrail WITH
      portainernumber = a
      pthberthr OF portainermasterportainer ← 1
      pthmsip ← FIRST OF berth[pthberthr OF portainermasterportainer]
      futportainershipnb OF portainermasterportainer ← shipnumber
      OF pthmsip
      futlocn OF portainermasterportainer ← a
      JOIN portainermasterportainer TO portainerper berth[1] IF totalload OF pthmsip > 0
      JOIN portainermasterportainer TO portaineractivelist IF totalload OF pthmsip > 0
      ACTIVATE portainermasterportainer FROM begin IN portainermod
    END
    IF portainermasterportainer IS NOT ACTIVE
  END
END
FOR a ← 7 TO 12
  portainermasterportainer ← FIRST portainer IN quayrail WITH
  portainernumber = a
  pthberthr OF portainermasterportainer ← 2
  futlocn OF portainermasterportainer ← a
  ACTIVATE portainermasterportainer FROM begin IN portainermod
  IF portainermasterportainer IS NOT ACTIVE
END
FOR a ← 13 TO 16
  portainermasterportainer ← FIRST portainer IN quayrail
  WITH portainernumber = a
  pthberthr OF portainermasterportainer ← 3
  futlocn OF portainermasterportainer ← a
  ACTIVATE portainermasterportainer FROM begin IN portainermod
  IF portainermasterportainer IS NOT ACTIVE
END
activeport ← 4
ACTIVATE agvmaster FROM begin IN agvmastmod
END
  IF big OF FIRST OF berth[2] = TRUE
    FOR a ← 3 TO 6
      portainermasterportainer ← FIRST portainer IN quayrail
      WITH portainernumber = a
      pthberthr OF portainermasterportainer ← 1
      pthmsip ← FIRST OF berth[pthberthr OF portainermasterportainer]
      futportainershipnb OF portainermasterportainer ← shipnumber
      OF pthmsip
      futlocn OF portainermasterportainer ← a
      JOIN portainermasterportainer TO portainerper berth[1] IF totalload OF pthmsip > 0
      JOIN portainermasterportainer TO portaineractivelist IF totalload OF pthmsip > 0
      ACTIVATE portainermasterportainer FROM begin IN portainermod
    END
    IF portainermasterportainer IS NOT ACTIVE
  END
END
PORTAINERMASTMOD - page 2

63  portainernumber = a
64  ptberthnr OF portainermasterportainer = 1
65  futlocnr OF portainermasterportainer = a
66  ACTIVATE portainermasterportainer FROM begin IN portainermod -
67  IF portainermasterportainer IS NOT ACTIVE
68  END
69  FOR a ← 7 TO 12
70    portainermasterportainer ← FIRST portainer IN quayrail WITH-
71    portainernumber = a
72    ptberthnr OF portainermasterportainer = 2
73    ptmship ← FIRST OF berth[ ptberthnr OF portainermasterportainer]
74    futportainershipnb OF portainermasterportainer ← shipnumber OF ptmsip
75    ptmship
76    futlocnr OF portainermasterportainer ← a
77    JOIN portainermasterportainer TO portainerperberth[ 2] IF →
78    totalload OF ptmship > 0
79    JOIN portainermasterportainer TO portaineractivelist IF-
80    totalload OF ptmship > 0
81    ACTIVATE portainermasterportainer FROM begin IN portainermod-
82    IF portainermasterportainer IS NOT ACTIVE
83  END
84  FOR a ← 13 TO 16
85    portainermasterportainer ← FIRST portainer IN quayrail WITH-
86    portainernumber = a
87    ptberthnr OF portainermasterportainer = 3
88    futlocnr OF portainermasterportainer ← a
89    ACTIVATE portainermasterportainer FROM begin IN portainermod-
90    IF portainermasterportainer IS NOT ACTIVE
91  END
92  activeport ← 6
93  ACTIVATE agvmaster FROM begin IN agvmastmod
94  END
95  END
98    FOR a ← 3 TO 6
99      portainermasterportainer ← FIRST portainer IN quayrail-
100         WITH portainernumber = a
101      ptberthnr OF portainermasterportainer ← 1
102      futlocnr OF portainermasterportainer ← a
103      ACTIVATE portainermasterportainer FROM begin IN-
104      portainermod IF portainermasterportainer IS NOT ACTIVE
105  END
106  FOR a ← 7 TO 12
107    portainermasterportainer ← FIRST portainer IN quayrail-
108        WITH portainernumber = a
109    ptberthnr OF portainermasterportainer ← 2
110    futlocnr OF portainermasterportainer ← a
111    ACTIVATE portainermasterportainer FROM begin IN portainermod-
112    IF portainermasterportainer IS NOT ACTIVE
113  END
114  FOR a ← 13 TO 16
115    portainermasterportainer ← FIRST portainer IN quayrail-
116        WITH portainernumber = a
117    ptberthnr OF portainermasterportainer ← 3
118    ptmship ← FIRST OF berth[ ptberthnr OF portainermasterportainer]
119    futportainershipnb OF portainermasterportainer ← shipnumber OF ptmship
120    futlocnr OF portainermasterportainer ← a
121    JOIN portainermasterportainer TO portainerperberth[ 3] IF-
122    totalload OF ptmship > 0
123    JOIN portainermasterportainer TO portaineractivelist IF-
124    totalload OF ptmship > 0
ACTIVATE portainermasterportainer FROM begin IN portainermod-
END
activeport = 4
ACTIVATE agvmaster FROM begin IN agvmastmod
END

FOR a ← 3 TO 6
portainermasterportainer ← FIRST portainer IN quayrail-
WITH portainernumber = a
ptberthnr OF portainermasterportainer ← 1
ptmship ← FIRST OF berth[ptberthnr OF portainermasterportainer]
futportainershipnb OF portainermasterportainer ← shipnumber OF ptmship
JOIN portainermasterportainer TO portainerper berth[1] IF →
totalload OF ptmship > 0
JOIN portainermasterportainer TO portaineractivilist IF →
totalload OF ptmship > 0
ACTIVATE portainermasterportainer FROM begin IN portainermod-
IF portainermasterportainer IS NOT ACTIVE
END
FOR a ← 7 TO 12
portainermasterportainer ← FIRST portainer IN quayrail WITH →
portainernumber = a
ptberthnr OF portainermasterportainer ← 2
ptmship ← FIRST OF berth[ptberthnr OF portainermasterportainer]
futportainershipnb OF portainermasterportainer ← shipnumber OF ptmship
futlocnr OF portainermasterportainer ← a
JOIN portainermasterportainer TO portainerper berth[2] IF →
totalload OF ptmship > 0
JOIN portainermasterportainer TO portaineractivilist IF →
totalload OF ptmship > 0
ACTIVATE portainermasterportainer FROM begin IN portainermod-
IF portainermasterportainer IS NOT ACTIVE
END
FOR a ← 13 TO 16
portainermasterportainer ← FIRST portainer IN quayrail →
WITH portainernumber = a
ptberthnr OF portainermasterportainer ← 3
futlocnr OF portainermasterportainer ← a
ACTIVATE portainermasterportainer FROM begin IN portainermod →
IF portainermasterportainer IS NOT ACTIVE
END
activeport ← 10
ACTIVATE agvmaster FROM begin IN agvmastmod
END

FOR a ← 3 TO 6
portainermasterportainer ← FIRST portainer IN quayrail-
WITH portainernumber = a
ptberthnr OF portainermasterportainer ← 1
futlocnr OF portainermasterportainer ← a
ACTIVATE portainermasterportainer FROM begin IN portainermod-
IF portainermasterportainer IS NOT ACTIVE
END
FOR a ← 7 TO 12
portainermasterportainer ← FIRST portainer IN quayrail WITH-
portainernumber = a
ptberthnr OF portainermasterportainer = 2
ptmship ← FIRST OF berth[ptberthnr OF portainermasterportainer]
furtportainershipnb OF portainermasterportainer ← shipnumber OF ptmship;
futlocnr OF portainermasterportainer ← a
JOIN portainermasterportainer TO portainerperberth[2] IF →
totalload OF ptmship > 0
JOIN portainermasterportainer TO portaineractivelist IF →
totalload OF ptmship > 0
ACTIVATE portainermasterportainer FROM begin IN portainermod →
IF portainermasterportainer IS NOT ACTIVE
END
FOR a ← 13 TO 16
portainermasterportainer ← FIRST portainer IN quayrail WITH→
portainernumber = a
ptberthnr OF portainermasterportainer ← 3
ptmship ← FIRST OF berth[ptberthnr OF portainermasterportainer]
furtportainershipnb OF portainermasterportainer ← shipnumber OF ptmship;
futlocnr OF portainermasterportainer ← a
JOIN portainermasterportainer TO portainerperberth[3] IF →
totalload OF ptmship > 0
JOIN portainermasterportainer TO portaineractivelist IF →
totalload OF ptmship > 0
ACTIVATE portainermasterportainer FROM begin IN portainermod →
IF portainermasterportainer IS NOT ACTIVE
END
activeport ← 10
ACTIVATE agvmaster FROM begin IN agvmastmod
END
END
FOR a ← 3 TO 6
portainermasterportainer ← FIRST portainer IN quayrail→
WITH portainernumber = a
ptberthnr OF portainermasterportainer ← 1
ptmship ← FIRST OF berth[ptberthnr OF portainermasterportainer]
furtportainershipnb OF portainermasterportainer ← shipnumber OF ptmship;
futlocnr OF portainermasterportainer ← a
JOIN portainermasterportainer TO portainerperberth[1] IF →
totalload OF ptmship > 0
JOIN portainermasterportainer TO portaineractivelist IF →
totalload OF ptmship > 0
ACTIVATE portainermasterportainer FROM begin IN portainermod →
IF portainermasterportainer IS NOT ACTIVE
END
FOR a ← 7 TO 12
portainermasterportainer ← FIRST portainer IN quayrail→
WITH portainernumber = a
ptberthnr OF portainermasterportainer ← 2
furtportainershipnb OF portainermasterportainer ← shipnumber OF ptmship;
futlocnr OF portainermasterportainer ← a
ACTIVATE portainermasterportainer FROM begin IN →
portainermod IF portainermasterportainer IS NOT ACTIVE
END
FOR a ← 13 TO 16
portainermasterportainer ← FIRST portainer IN quayrail →
WITH portainernumber = a
ptberthnr OF portainermasterportainer ← 3
ptmship ← FIRST OF berth[ptberthnr OF portainermasterportainer]
furtportainershipnb OF portainermasterportainer ← shipnumber OF ptmship;
futlocnr OF portainermasterportainer ← a
JOIN portainermasterportainer TO portainerperberth[3] IF →
totalload OF ptmship > 0
JOIN portainermasterportainer TO portaineractivelist IF-
  totalload OF ptmship > 0
ACTIVATE portainermasterportainer FROM begin IN portainermod-
  IF portainermasterportainer IS NOT ACTIVE
END
activeport ← 8
ACTIVATE agvmaster FROM begin IN agvmastmod
END

FOR a ← 3 TO 6
  portainermasterportainer ← FIRST portainer IN quayrail-
  WITH portainernumber = a
  ptberthnr OF portainermasterportainer ← 1
  ptmship ← FIRST OF berth[ptberthnr OF portainermasterportainer]
  futportainershipnb OF portainermasterportainer ← shipnumber OF ptmship
  futlnocr OF portainermasterportainer ← a
  JOIN portainermasterportainer TO portainerperberth[1] IF-
  totalload OF ptmship > 0
  JOIN portainermasterportainer TO portaineractivelist-
  IF totalload OF ptmship > 0
  ACTIVATE portainermasterportainer FROM begin IN portainermod-
  IF portainermasterportainer IS NOT ACTIVE
END

FOR a ← 7 TO 12
  portainermasterportainer ← FIRST portainer IN quayrail-
  WITH portainernumber = a
  ptberthnr OF portainermasterportainer ← 2
  ptmship ← FIRST OF berth[ptberthnr OF portainermasterportainer]
  futportainershipnb OF portainermasterportainer ← shipnumber OF ptmship
  futlnocr OF portainermasterportainer ← a
  JOIN portainermasterportainer TO portainerperberth[2] IF-
  totalload OF ptmship > 0
  JOIN portainermasterportainer TO portaineractivelist IF-
  totalload OF ptmship > 0
  ACTIVATE portainermasterportainer FROM begin IN portainermod-
  IF portainermasterportainer IS NOT ACTIVE
END

FOR a ← 13 TO 16
  portainermasterportainer ← FIRST portainer IN quayrail-
  WITH portainernumber = a
  ptberthnr OF portainermasterportainer ← 3
  ptmship ← FIRST OF berth[ptberthnr OF portainermasterportainer]
  futportainershipnb OF portainermasterportainer ← shipnumber OF ptmship
  futlnocr OF portainermasterportainer ← a
  JOIN portainermasterportainer TO portainerperberth[3] IF-
  totalload OF ptmship > 0
  JOIN portainermasterportainer TO portaineractivelist IF-
  totalload OF ptmship > 0
  ACTIVATE portainermasterportainer FROM begin IN portainermod-
  IF portainermasterportainer IS NOT ACTIVE
END
activeport ← 14
ACTIVATE agvmaster FROM begin IN agvmastmod
END
PASSIVATE
pal is portainer active list

This means there are portainers active

if actlocnr is not futlocnr the portainer is not yet at the right location. From the moment
the portainer is at its right location it can get rgv's assigned again

In line 20 to 40 the agvmaster looks what the portainers are doing and determines the
number of needed rgv's for the portainers

From line 46 to 77 the right number of rgv's per portainer are activated

Bye is an attribute of the rgy of the type logical. If bye of a rgy is true it has to go to
the parking the moment it has finished the assignment it is working on

In line 54 to 77 extra rgy's are created
1 begin:
2 @determine the number of active portainers
3
4 portaineropplek:
5 pal ← LENGTH OF portaineractivelist
6 IF LENGTH OF portaineractivelist = 0
7     GOTO beter2
8 END
9
10 WHILE pal ≠ 0
11 agvpportainer-FIRST portainer IN portaineractivelist
12 REMOVE agvpportainer FROM portaineractivelist
13 JOIN agvpportainer TO portaineractivelist
14
15 IF actlocnr OF agvpportainer ≠ futlocnr OF agvpportainer
16     agvppneeded−0
17     GOTO beter
18 END
19 IF ptship OF agvpportainer BELONGS TO unloadingship & actlocnr OF→
20     agvpportainer = futlocnr OF agvpportainer
21     agvppneeded−6 @ even tijdelijk aagepast
22     pp ← agvppneeded − (LENGTH OF agvvppactive OF agvpportainer +→
23     LENGTH OF agvvppactive OF agvpportainer)
24 END
25 IF ptship OF agvpportainer BELONGS TO loadingship & actlocnr OF→
26     agvpportainer = futlocnr OF agvpportainer & firstshuffleagv OF
27     ptship OF agvpportainer = 1
28     agvppneeded−12 @ even tijdelijk aagepast
29     pp−agvppneeded − (LENGTH OF agvvppactive OF agvpportainer +→
30     LENGTH OF agvvppactive OF agvpportainer)
31 END
32 IF ptship OF agvpportainer BELONGS TO loadingship & actlocnr OF→
33     agvpportainer = futlocnr OF agvpportainer & firstshuffleagv OF
34     ptship OF agvpportainer ≠ 1
35     @GOTO beter
36     agvppneeded−8 @ even tijdelijk aagepast
37     pp−agvppneeded − (LENGTH OF agvvppactive OF agvpportainer +→
38     LENGTH OF agvvppactive OF agvpportainer)
39 END
40
41 IF pp = 0 @er zijn genoeg agv's
42 GOTO beter
43 END
44
45 IF pp<0 @er zijn teveel agv's
46 FOR i−1 TO (−1*pp)
47     byeagv ← FIRST agv IN agvvppactive OF agvpportainer
48     REMOVE byeagv FROM agvvppactive OF agvpportainer
49     JOIN byeagv TO agvvppactive OF agvpportainer
50     bye OF byeagv ← TRUE
51 END
52
53 IF pp>0 @er zijn te weinig agv's
54 FOR i−1 TO pp
55     IF LENGTH OF agvvppactive OF agvpportainer=0
56         hiatagv-FIRST agv IN agvparking WITH (agvnumber)MODULO(2)=1
57         JOIN hiatagv TO agvvppactive_odd OF agvpportainer
58         REMOVE hiatagv FROM agvparking
59     GOTO vreter
60 END
61
In line 66 to 76 it is made sure there are just as much odd as even rgv's. If \( ri/ra > 0.5 \) this means there are more odd rgv's so an even rgv must be activated.

If \( ri/ra < 0.5 \) this means there are more even rgv's so an odd rgv must be activated.

In line 78 to 100 the start conditions for the rgv's are created.

The starting position of the rgv is chosen.

From line 105 to the end is a separate part of the agv master which has nothing to do with the precedent.
WHILE LENGTH OF agvppactive_odd OF agvportainer>0
  ri-LENGTH OF agvppactive_odd OF agvportainer
  ra-LENGTH OF agvppactive OF agvportainer
  IF (ri/ra)>0.5
    HIAGV-FIRST agv IN agvparking WITH (agvnumber) MODULO(2)=0
    REMOVE hiagv FROM agvparking
    GOTO vreter
  END
  IF (ri/ra)\leq0.5
    hiagv- FIRST agv IN agvparking WITH (agvnumber) MODULO(2)=1
    JOIN hiagv TO agvppactive_odd OF agvportainer
    REMOVE hiagv FROM agvparking
    GOTO vreter
  END
END

vrer:
  @REMOVE FIRST agv IN agvparking FROM agvparking
  JOIN hiagv TO activeagvlist
  JOIN hiagv TO agvworklist
  JOIN hiagv TO agvppactive OF agvportainer
  agvberthnb OF hiagv = shipberthnumber OF ptshelf OF agvportainer
  yard OF hiagv=TRUE
  ro-LENGTH OF agvppactive OF agvportainer
  ru-CEIL(ro/2)
  ro-(ru) MODULO(2)
  IF ro=0
    cont20shuf OF hiagv = TRUE
  END
  IF ro =1
    cont20shuf OF hiagv = FALSE
  END
  x OF hiagv = futlocnr OF agvportainer
  y OF hiagv= CEIL((x OF hiagv)/2)*20 - 18 +
  ((-1 + (x OF hiagv)) MODULO(2))*9
  yy OF hiagv = y OF hiagv
  ACTIVATE hiagv FROM begin IN agvmod
END

beter:
  pal-pal-1
END

beter2:
  WAIT
  byeshuffle:
  amship = LAST ship IN activeshiplist
IF LENGTH OF agvberthshufflenlist[shipberthnumber OF amship]>0
  FOR i=1 TO LENGTH OF agvberthshufflenlist[shipberthnumber OF amship]
    byeagv = FIRST agv IN agvberthshufflenlist[shipberthnumber OF amship]
    bye OF byeagv-TRUE
  END
END

PASSIVATE
Line 3 to 70 is about the positioning of the portainer at the quay

Ptportloc is the portainer location the portainer occupies at the moment

if the length of assignedrgv's of the portainer location > 0 means there is a rgv on its way to be served by this particular portainer at that particular place. So it is for the portainer impossible to move to another location it will first have to serve this rgv

In contrast to line 11 there are no rgv's assigned. The portainer looks weather it is at the right location (inpositioncheck).

If in line 23 the condition counts the portainer is at the right location

If the condition in line 41 counts the portainer has to move to a portlocation with a higher number. The portainer waits until the adjacent place is free

If the condition in line 56 counts the portainer has to move to a portlocation with a lower number
PORTAINERMOD - page 1

1 begin:
2
3 @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
4 @getting in position of the portainers@
5 @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

6 ptporthloc <- FIRST portloc IN portlocations WITH portlocnumber->
7 = actlocnr OF THIS portainer
8 quaylocation[ actlocnr ] <- TRUE
9
10 IF LENGTH OF assignedagvs OF ptporthloc ≠ 0
11 asagv <- FIRST agv IN assignedagvs OF ptporthloc
12 IF asagv IS NONE
13 GOTO impositioncheck
14 END
15 ENTER locationset OF ptporthloc IF THIS portainer->
16 BELONGS NOT TO locationset OF ptporthloc
17 GOTO restart
18 END

19 IF LENGTH OF assignedagvs OF ptporthloc = 0
20
21 impositioncheck:
22 IF futlocnr OF THIS portainer = actlocnr OF THIS portainer
23 ptporthloc <- FIRST portloc IN portlocations WITH->
24 portlocnumber = futlocnr OF THIS portainer
25 ptship <- FIRST OF berth[ ptberthnr]
26 IF ptship IS NONE
27 PASSIVATE
28 END
29 futportainershipnb <- shipnumber OF ptship
30 ENTER locationset OF ptporthloc IF THIS portainer ->
31 BELONGS NOT TO locationset OF ptporthloc
32 IF THIS portainer BELONGS TO portaineractivelist &->
33 LENGTH OF agvppactive OF THIS portainer= 0
34 WORK 1 SECOND
35 ACTIVATE agvmast FROM portaineropplek IN agvmastmod->
36 IF agvmaster IS NOT ACTIVE
37 END
38 GOTO restart
39 END

40 IF futlocnr OF THIS portainer > actlocnr OF THIS portainer
41 WAIT UNTIL quaylocation[ actlocnr+1 ] = FALSE
42 ptporthloc <- FIRST portloc IN portlocations WITH-
43 portlocnumber = actlocnr
44 LEAVE locationset OF ptporthloc IF THIS portainer->
45 BELONGS TO locationset OF ptporthloc
46 quaylocation[ actlocnr ] <- FALSE
47 actlocnr OF THIS portainer <- (actlocnr OF THIS portainer+1)
48 ptporthloc <- FIRST portloc IN portlocations WITH ->
49 portlocnumber = actlocnr
50 ENTER locationset OF ptporthloc IF THIS portainer->
51 BELONGS NOT TO locationset OF ptporthloc
52 quaylocation[ actlocnr ] <- TRUE
53 GOTO impositioncheck
54 END

55 IF futlocnr OF THIS portainer < actlocnr OF THIS portainer
56 WAIT UNTIL quaylocation[ actlocnr-1 ] = FALSE
57 ptporthloc <- FIRST portloc IN portlocations WITH-
58 portlocnumber = actlocnr
59 LEAVE locationset OF ptporthloc IF THIS portainer->
60 BELONGS TO locationset OF ptporthloc
61 quaylocation[ actlocnr ] <- FALSE
From line 72 the actual loading and unloading of the rgv's starts

**Portlocqueue** is a set containing rgv's waiting under the portainer to be served

**Portasqueue** is a set containing rgv's assigned to that portlocation. These rgv's could be
empty wanting a container from the portainer

**Portainertimes** is a macro creating the portainercycletimes

qq is the portainercycletime calculated in portainertimes

**Line 75 to 81** a rgv is assigned to the portainer this does not automatically means the
rgv is already at the portainerlocation it could be on the way

**Line 83 to 90** a rgv has arrived under the portainer in the queue

**Line 92 to 109** deals with the unloading of the vessel

The portainer has already worked qq seconds to get the container from the vessel
the 10 seconds in line 98 are for the actual positioning of the container on the rgv

The portainer removes the rgv from its queue and prepares the rgv to leave the
portainerlocation

**Line 110 to 135** deal with the loading of the vessels

The portainer removes the container from the rgv this takes 10 seconds hereafter the
rgv is able to move on while the portainer has to do its work

In line 120 the portainer keeps count so it knows when the vessel can leave the berth

**Line 122 to 129** prepare the rgv to leave the portainerlocation
actlocnr OF THIS portainer ← (actlocnr OF THIS portainer - 1)
ptportloc ← FIRST portloc IN portlocations WITH →
portlocnumber ← actlocnr
ENTER locationset OF ptportloc IF THIS portainer→
BELONGS NOT TO locationset OF ptportloc
quaylocation[ actlocnr ] ← TRUE
GOTO inpositioncheck
END

RESTART:
WAIT WHILE portlocqueue OF ptportloc IS EMPTY & LENGTH→
OF portasqueue OF THIS portainer = 0
IF LENGTH OF portasqueue OF THIS portainer ≠ 0
asagv ← FIRST agv IN portasqueue OF THIS portainer
JUMP TO portainertimes
WAIT (QQ OF THIS portainer) SECONDS
REMOVE asagv FROM portasqueue OF THIS portainer
WAIT WHILE portlocqueue OF ptportloc IS EMPTY
END

asagv ← FIRST agv IN portlocqueue OF ptportloc→
WITH x = actlocnr OF THIS portainer
IF loadedagv OF portagv = TRUE
GOTO load @ is het laden van het schip
END
IF loadedagv OF portagv = FALSE
GOTO unload
END

@the unloading of the vessel@  
@the unloading of the vessels@  

IF portlocqueue OF ptportloc IS NOT EMPTY
WORK 10 SECONDS
loadedagv OF portagv ← TRUE
REMOVE portagv FROM portlocqueue OF ptportloc
REMOVE portagv FROM assignedagvs OF ptportloc→
IF portagv BELONGS TO assignedagvs OF ptportloc
ACTIVATE portagv FROM begin IN agvmod
JOIN portagv TO agvworklist @ deze zit er nieuw in
END
IF assignedagvs OF ptportloc IS EMPTY
GOTO inpositioncheck
END

@the loading of the vessels@  

IF loadedagv OF portagv = TRUE
IF portlocqueue OF ptportloc IS NOT EMPTY
WORK 10 SECONDS
totalload of ptship ← (totalload of ptship) - 1
loadedagv OF portagv ← FALSE
REMOVE portagv FROM portlocqueue OF ptportloc
REMOVE portagv FROM assignedagvs OF ptportloc→
portagv BELONGS TO assignedagvs OF ptportloc

This is where the portainer works the container to its place on/in the vessel. If totalload = 0, this means the vessel is completely loaded and ready to leave. The portainer that has delivered the last container tells the vessel to leave.
JOIN portagv TO agvworklist
yard OF portagv = FALSE
ACTIVATE portagv FROM begin IN agvmmod
asagy = FIRST agv IN assignedagvs OF ptportloc
  IF totalload OF ptship = 0
    GOTO ietstevroeg
  END
  JUMP TO portainertimes
  WORK (qq OF THIS portainer) SECONDS
END

IENTSTEVROEG:
  IF totalload OF ptship = 0
    ACTIVATE ptship FROM leaving IN shipmod IF ptship IS NOT ACTIVE
  END
  IF assignedagvs OF ptportloc IS EMPTY
    GOTO inpositioncheck
  END
  GOTO begin

"
the exportgen creates the exportcontainers for the vessels

If big is true this vessel is a mega vessel

If big is false this vessel is a feeder

totalload is the number of containers the vessel has to receive at the terminal here
this number is deterministic. Totalload starts at the maximum when it is 0 the vessel can
leave the terminal
totalu is the start value of totalload, this value doesn't change in contrast to totalload

propex40 is the share of 40 foot containers
begin:

exportship ← THIS ship 0FIRST ship IN anchorage
IF big OF exportship = TRUE
ww←1575
END
IF big OF exportship = FALSE
ww←525
END

totalload OF exportship ← ww
totalu OF exportship ← ww
ex40 OF exportship ← 0.5*ww
ex20 OF exportship ← 0.5*ww

propex40 ← ex40 OF exportship/totalload OF exportship
propex20 ← ex20 OF exportship/totalload OF exportship

ACTIVATE exportship FROM berthpermission IN shipmod IF exportship IS NOT ACTIVE

WAIT
The importgen creates the import containers for the vessels
begin:
importship = THIS ship @FIRST ship IN anchorage
IF big OF importship = TRUE
    ww=18
END
IF big OF IMPORTSHIP = FALSE
    ww=20
END

totalunload OF importship = ww
total OF importship = ww
im40 OF importship = 0.5*ww
im20 OF importship = 0.5*ww

propim40 OF importship = (im40 OF importship)/(totalunload OF importship)
propim20 OF importship = (im20 OF importship)/(totalunload OF importship)

ACTIVATE importship FROM export IN shipmod IF IMPORTSHIP IS NOT ACTIVE
WAIT
The first thing for the rgv is to activate the trafficmaster and ask for a assignment. The trafficmaster will looking at the attributes of the rgv activate the rgv from a specific point in the agv module. The trafficmaster activates the rgv from shuffling, toyard or uneasy etc. it all starts in line 8 to 10 proclaimnodes is a macro that looks whether it is possible for the rgv to enter the main track and start driving. This macro enables the rgv to enter the main track without interfering with other rgv's. The rgv only returns from this macro when the main track the rgv wants to enter is not claimed by another rgv line 15 the rgv returns from the macro and claims the entrance node extime is a specific time for a rgv to pull up and leave a node wheel rotation of the rgv takes 4 seconds This is the driving on the node line 25 entrance node is not needed anymore and removed from the set entrance line 25 exit is the first node where the rgv has to leave the main track it has just entered in line 31 to 53 the actual driving of the rgv takes place this is a great loop line 32 side_track is a macro, this can be a macro "short cut" or "stack_cran" or "super shortcut" in line 32 to 38 there are six the same statements concerning the side_track this is because it is possible a rgv has to take a few short cuts in a row line 38 to 43 concerns the driving on the main track line 42 is the driving of the current section minus the time needed for accelerating and slowing down these times are already deducted in line 40 and 46
begin:
WAIT WHILE trafficmaster IS ACTIVE
ACTIVATE trafficmaster FROM begin IN trafficmastmod IF trafficmaster IS NOT ACTIVE
WAIT

shuffling:
toyard:
oneasy:
toquay:
go-ahead:

entrance_node ← FIRST node IN entrance OF THIS agv
JUMP TO preclaimnodes

claimed OF entrance_node ← claimed OF entrance_node+1
cur_node←entrance_node
WORK exittime OF cur_node

LEAVE stack_row OF agvstack OF THIS agv

WORK 8 SECONDS

WORK (nodedistance1 OF cur_node)/(nodespeed OF cur_node)

REMOVE entrance_node FROM entrance OF THIS agv
exit ← FIRST node IN route OF THIS agv
REMOVE FIRST node IN route OF THIS agv FROM route OF THIS agv

op_naar_het_stack:
JUMP TO side_track[1] OF cur_node IF cur_node IS exit
JUMP TO side_track[1] OF cur_node IF cur_node IS exit
JUMP TO side_track[1] OF cur_node IF cur_node IS exit
JUMP TO side_track[1] OF cur_node IF cur_node IS exit
JUMP TO side_track[1] OF cur_node IF cur_node IS exit
JUMP TO side_track[2] OF cur_node IF cur_node IS superexit
cur_sec←outs[1] OF cur_node
WORK (nodedistance1 OF cur_node)/(nodespeed OF cur_node)
claimed OF cur_node←claimed OF cur_node-1
WORK MAX(0,(s_length OF cur_sec-(nodedistance1 OF
cur_node+nodedistance2 OF cur_node))/(nodespeed OF cur_node))
lastnode OF THIS agv ← cur_node
cur_node←end_point OF cur_sec
claimed OF cur_node←claimed OF cur_node+1

WORK (nodedistance2 OF cur_node)/(nodespeed OF cur_node)
JUMP TO side_track[1] OF cur_node IF cur_node IS exit
JUMP TO side_track[1] OF cur_node IF cur_node IS exit
JUMP TO side_track[1] OF cur_node IF cur_node IS exit
JUMP TO side_track[1] OF cur_node IF cur_node IS exit
JUMP TO side_track[2] OF cur_node IF cur_node IS superexit

REPEAT FROM op_naar_het_stack
gaan_in_de_rij_staan:
ACTIVATE agvcrane FROM begin IN cranemod IF agvcrane IS NOT ACTIVE
WAIT

toparking:
JUMP TO agv_toparking
Agyworklist a set containing all the rgv's requesting the trafficmaster for work

If the agv has decided the rgv has to go because there are to many rgv's active he
will give an attribute bye, this is spotted by the trafficmaster and he will send the rgv to
the parking line 6 to 10

If shuffleagv is true it means the rgv is shuffling and does not have to work for a
portainer any more it has to be removed from pactive

This is the case when a rgv is working for a portainer that has just finished a vessel
and the rgv does not have to shuffle

The rgv has asked the trafficmaster for an assignment but the portainer this rgv is
working for is not at the right location

In line 38 to 76 a rgv calls from the yard and the vessel is unloading

If the conditions in line 42 are correct the trafficmaster identifies that the rgv calls from the
yard and the vessel it is working for is unloading

In line 48 to 54 the totalunload =0 the vessel is unloaded and has to be loaded

If inplacecounter >0 this would mean the vessel can be loaded with containers from the
same berth. This is possible if some shuffling is done beforehand

agv_load_back is a macro determining the route the rgv has to take from the yard
to the right portainerlocation

The rgv will go to the portainerlocation and is assigned, this enables the portainer to
have a container ready when the rgv arrives.
begin:
tmagv ← FIRST agv IN agworklist
tmship ← FIRST ship IN activeshiplist WITH shipberthnumber = agvberthnb OF tmagv
tmport ← FIRST portainer IN quayrail WITH actlocrn = x OF tmagv

IF BYE OF TMAGV = TRUE & LOADEDAGV OF TMAGV = FALSE
  REMOVE TMAGV FROM AGWORKLIST
  ACTIVATE TMAGV FROM TOPARKING IN AGVMOD
  WAIT
END

IF shuffleagv OF tmagv = TRUE
  REMOVE tmagv FROM agyppactive OF tmport
  JOIN tmagv TO agyppactive OF tmport
  GOTO shuffleprocedure
END

IF tmship IS NONE
  ACTIVATE tmagv FROM toparking IN agvmod IF tmagv IS NOT ACTIVE
  REMOVE tmagv FROM agworklist
  WAIT
END

IF tmport IS NONE
  ACTIVATE tmagv FROM toparking IN agvmod IF tmagv IS NOT ACTIVE
  REMOVE tmagv FROM agworklist
  loadedagv OF tmagv ← FALSE
  WAIT
END

IF actlocrn OF tmport ≠ futlocrn OF tmport & loadedagv OF tmagv = FALSE
  ACTIVATE tmagv FROM toparking IN agvmod IF tmagv IS NOT ACTIVE
  REMOVE tmagv FROM agworklist
  WAIT
END

restart:

@ empty agv calls from the yard ship unloading@

IF yard OF tmagv = TRUE & loadedagv OF tmagv = FALSE & tmship BELONGS TO unloadingship
totalunload OF tmship = totalunload OF tmship - 1
REMOVE tmship FROM unloadingship
JOIN tmship TO unloadingship

IF totalunload OF tmship = 0
  REMOVE tmship FROM unloadingship
  JOIN tmship TO loadingship
  IF inplacecounter[ shipberthnumber OF tmship] ≤ 0
    ACTIVATE agymaster FROM begin IN agvmastmod IF tmagv IS NOT ACTIVE
END
END

yard OF tmagv ← FALSE
JUMP TO rgv_unload_back

agvportloc ← FIRST portloc IN portlocations WITH portlocnumber = x OF tmagv
JOIN tmagv TO assignedagys OF agvportloc IF tmagv BELONGS NOT TO assignedagys OF agvportloc
JOIN tmagv TO portasqueue OF tmport IF tmagv BELONGS NOT TO portasqueue
Line 74 to 114 deals with empty rgv's calling from the yard when there is a vessel loading.

_tmtotalx_ is the number of containers that still have to go with the vessel.

If the condition in line 83 counts all the containers are on board of the vessel and the shuffling at the terminal can start.

In line 88 to 96 the rgv is being prepared for the shuffling.

_Loadyardallocation_ is a macro that chooses where the rgv has to get a load container.

_agv_stack_stack_ is a macro that tells the rgv how to get to the right stack (route).

The rgv is being assigned to the portlocation because it has a assignment to eventually deliver a container to a portainer so it is important the portainer knows the rgv is coming.

In line 116 to 137 a loaded rgv calls from the yard and its vessel is loading.
OF tmport
ACTIVATE tmport FROM restart IN portainermod IF tmport IS NOT ACTIVE
IF tmagv IS ACTIVE
    CANCEL tmagv
END
ACTIVATE tmagv FROM toquay IN agvmod IF tmagv IS NOT ACTIVE
REMOVE tmagv FROM agvworklist
WAIT
END

@empty calls from the yard and ship loading @

IF yard OF tmagv = TRUE & loadedagv OF tmagv = FALSE & tmship BELONGS TO loadingship
    tmtoalex ← (ex40 OF tmship + ex20 OF tmship)
    tex OF tmship ← MAX(0, (ex40 OF tmship + ex20 OF tmship))
END

IF (tex OF tmship = 0 | tmtoalex = 0) @overgang naar shuffelen
    ACTIVATE tmagv FROM toparking IN agvmod IF tmagv IS NOT ACTIVE
    REMOVE tmagv FROM agvworklist
    loadedagv OF tmagv ← FALSE
    WAIT
    shuffleagv OF tmagv ← TRUE
    easyload OF tmagv ← FALSE
    firstshuffle OF tmagv ← TRUE
    IF firstshuffleagv OF tmship = 1
        ACTIVATE agvmaster FROM begin IN agvmastmod
        firstshuffleagv OF tmship ← (firstshuffleagv OF tmship) - 1
    END
    JOIN tmagv TO agvberthshuffelist[agvberthnb OF tmagv]
    GOTO shuffleprocedure
END

JUMP TO loadyardallocation
JUMP TO agv_stack_stack
agvportloc ← FIRST portloc IN portlocations WITH portlocnumber = actlocnr OF tmport
JOIN tmagv TO assignedagvs OF agvportloc IF tmagv→ BELONGS NOT TO assignedagvs OF agvportloc

IF tmagv IS ACTIVE
    CANCEL TMAGV
END

ACTIVATE tmagv FROM toyard IN agvmod
REMOVE tmagv FROM agvworklist
WAIT
END

@loaded agv calls from the yard with ship loading@

IF yard OF tmagv = TRUE & loadedagv OF tmagv = TRUE & tmship BELONGS TO loadingship
    yard OF tmagv ← FALSE
    again2:
    tmship ← FIRST ship IN activeshiplist WITH→
agv_stack_berth is a macro that creates the route from the stack to the right berth

Line 138 to 157 is about an empty rgv that calls from the quay with a vessel loading container

unloadyardallocation is a macro that decides where the rgv has to deliver the container

agv_load creates the route for the rgv to get to the stack

In line 171 to 184 if the condition counts it would mean the vessel is loaded and the rgv has got to be prepared for the shuffling
125  _shipberthenumber=agvberthenb OF tmagv
126  JUMP TO agv_stack_berth
127
128  REMOVE tmagv FROM agvworklist
129  IF tmagv IS ACTIVE
130    CANCEL tmagv
131  END
132  IF TMAGV IS ACTIVE
133    CANCEL TMAGV
134  END
135  ACTIVATE tmagv FROM toquay IN agvmod IF tmagv IS NOT ACTIVE
136  WAIT
137 END
138  @loaded agv calls from portainer ship is unloading
139  @empty calls from the quay and ship is loading
140  IF yard OF tmagv = FALSE & loadedagv OF tmagv = TRUE
141    & tmship BELONGS TO unloadingship
142    tmship ← FIRST ship IN activeshiplist WITH
143    berthenumber=agvberthenb OF tmagv
144    yard OF tmagv ← TRUE
145    JUMP TO unloadyardallocation
146
147  JUMP TO agv_unload
148  IF TMAGV IS ACTIVE
149    CANCEL TMAGV
150  END
151  ACTIVATE tmagv FROM toyard IN agvmod IF tmagv IS NOT ACTIVE
152  REMOVE tmagv FROM agvWorklist
153  WAIT
154 END
155
156  IF yard OF tmagv = FALSE & loadedagv OF tmagv = FALSE
157    & tmship BELONGS TO loadingship
158  again5:
159    tmship ← FIRST ship IN activeshiplist WITH
160    berthenumber=agvberthenb OF tmagv
161  from1:
162  tmtotalex ← (ex40 OF tmship + ex20 OF tmship)
163  tex OF tmship ← MAX(0, (ex40 OF tmship + ex20 OF tmship))
164
165  IF (tex OF tmship = 0 | tmtotalex = 0)
166    ACTIVATE tmagv FROM toparking IN agvmod IF tmagv IS NOT ACTIVE
167    REMOVE tmagv FROM agvworklist
168    loadedagv OF tmagv ← FALSE
169    WAIT
170  shuffleagv OF tmagv ← TRUE
171  firstshuffle OF tmagv ← TRUE
172  easylag OF tmagv ← FALSE
173  IF firstshuffleagv OF tmship = 1
174    ACTIVATE agvmaster FROM begin IN agvmasmod
175    firstshuffleagv OF tmship ← (firstshuffleagv OF tmship) - 1
176 END
177  JOIN tmagv TO agvberthshuffeulist( agvberthenb OF tmagv)
178  GOTO shuffleprocedure
Loadyard allocation is a macro that chooses where the rgv has to get a load container

agv_stack_berth is a macro that creates the route from the stack to the right berth

In line 200 to 213 the right stackcrane is determined

This formula makes it possible to vary the number of cranes at the terminal

230 to 249 deals with a loaded rgv calling from the portainer and its vessel is loading

This looks like a strange situation, this rgv has just received the last unload container and now asks the trafficamster what to do. This is the transition between unloading and loading

agv_tossen creates the route for the rgv to get to the stack
187 JUMP TO loadyardallocation
188
189 IF (agvnumber OF tmagv)MODULO(2) = 0 & (uneasy OF tmagv = TRUE)
190 JUMP TO agv_uneasy_b_s
191 GOTO skip
192 END
193
194 JUMP TO agv_berth_stack
195
196 skip:
197
198 REMOVE tmship FROM loadingship
199 JOIN tmship TO loadingship
200 yyyy OF tmagv-y OF tmagv
201 xxx OF tmagv-FLOOR(((yyyy OF tmagv-1)/20)
202 zzz OF tmagv-FLOOR(((yyyy OF tmagv-1)/20)
203 yyyy OF tmagv-(y OF tmagv)-(zzz OF tmagv)*20
204
205 IF (yyyy OF tmagv)MODULO(2)=0
206 yyyy OF tmagv -= yyyy OF tmagv -1
207 END
208 @IF (yyyy OF tmagv) > 10
209 @yyyy OF THIS agv-= yyyy OF tmagv -10
210 @END
211
212 yyyy OF tmagv -= yyyy OF tmagv + 20*(xxx OF tmagv)
213 tmcrane = FIRST crane IN cranelist WITH cranelistnumber = yyyy OF tmagv
214 JOIN tmagv TO assignedqueue OF tmcrane
215
216 REMOVE tmcrane FROM cranelist
217 JOIN tmcrane TO cranelist
218 REMOVE tmagv FROM agvworklist
219 agvportloc = FIRST portloc IN portlocations WITH-
220 portlocnumber = actionn OF tmporpt
221 JOIN tmagv TO assignedagvs OF agvportloc IF →
222 tmagv BELONGS NOT TO assignedagvs OF agvportloc
223 IF TMAGV IS ACTIVE
224 CANCEL TMAGV
225 END
226 ACTIVATE tmagv FROM toyard IN agvmod IF tmagv IS NOT ACTIVE
227 WAIT
228 END
229
230 @ложенй agv calls from the portainer and ship is loading, dit is de overgang@
231 @ложенй agv calls from the portainer and ship is loading, dit is de overgang@
232
233 IF yard OF tmagv = FALSE & loadedagv OF tmagv =TRUE→
234 & tmship BELONGS TO loadingship
235 JUMP TO unloadyardallocation
236 JUMP TO agv_unload
237 IF TMAGV IS ACTIVE
238 CANCEL TMAGV
239 END
240 ACTIVATE tmagv FROM toyard IN agvmod IF tmagv IS NOT ACTIVE
241 yard OF tmagv-TRUE
242 REMOVE tmagv FROM agvworklist
243 WAIT
244
245 END
246 WAIT
From line 250 to the end has everything to do with the shuffling at the terminal

an empty rgv is under its portainer or crane and about to retrieve a container from the stack behind the portainer for the first time.

yardshuffleallocation is a macro that decides where to collect a shuffle container

If this condition counts there are no more containers to be shuffled and the rgv can go to the parking

If this condition counts there are shuffle containers and one is deducted

odd rgv's take a different route than even rgv's

agv_berth_stack creates the route from the berth to the stack

agv_uneasy_b_s creates a route for the rgv from berth to stack

y_creation is a macro responsible for determining the right stacking crane as in line 200

empty rgv under a crane and about to retrieve a container from the current stack behind the portainer from line 306 to 340
shuffleprocedure:

IF shuffleavg OF tmavg = TRUE

IF loadedavg OF tmavg = FALSE & (CEIL((y OF tmavg)/10)=x OF tmavg) | yard OF tmavg = FALSE)
& firstshuffle OF tmavg = TRUE

JUMP TO yardshuffleallocation

IF fromto(t[agvberthnb OF tmavg,sbnr OF tmavg]=0
  bye OF tmavg = TRUE
END
IF fromto(t[agvberthnb OF tmavg,sbnr OF tmavg]>0
  fromto(t[agvberthnb OF tmavg,sbnr OF tmavg]=fromto->
  [agvberthnb OF tmavg,sbnr OF tmavg]=-1
END

IF yard OF tmavg = FALSE
  IF (agvnumber OF tmavg)MODULO(2)=1
    JUMP TO agv_berth_stack @receiving a route
  END
  IF (agvnumber OF tmavg)MODULO(2)=0 ->
    & CEIL((yy OF tmavg)/60)≠(agvberthnb OF tmavg)
    JUMP TO agv_uneasy_b_s @receiving a route
  END
  IF (agvnumber OF tmavg)MODULO(2)=0 ->
    & CEIL((yy OF tmavg)/60)= (agvberthnb OF tmavg)
    JUMP TO agv_berth_stack @receiving a route
  END

@- - A new agv is generated to help shuffling -@-
IF yard OF tmavg = TRUE
  easyload OF tmavg = TRUE
  JUMP TO agv_stack_stack
  easyload OF tmavg = FALSE
END
JUMP TO y_creation @nieuw
JOIN tmavg TO assignedqueue OF agvcrane OF tmavg IF tmavg→
  BELONGS NOT TO assignedqueue OF agvcrane
IF TMAVG IS ACTIVE
  CANCEL TMAVG
END
ACTIVATE tmavg FROM shuffling IN agvmid IF tmavg IS NOT ACTIVE
REMOVE tmavg FROM agworklist
WAIT
END

@- agv is empty and is located under its crane and about to retrieve -@
@-- a container from the current stack behind the portainer --@
@--- agv brought a container from a strange berth ---@
IF loadedagv OF tmagv = FALSE & CEIL((y OF tmagv)/10)=(x OF tmagv)
& yard OF tmagv = TRUE & sbnr OF tmagv = 0

IF bye OF tmagv = TRUE
GOTO bye
END
JUMP TO yardshuffeleallocation @receiving sbnr

IF fromtob( agvberthnb OF tmagv,sbnr OF tmagv)>0
tmcrane = FIRST crane IN cranelist WITH cranenumber = yy OF tmagv
fromtob( agvberthnb OF tmagv,sbnr OF tmagv)-fromtob( agvberthnb-
OF tmagv,sbnr OF tmagv)-1
JOIN tmagv TO cranequeue OF tmcrane IN FRONT POSITION
REMOVE tmagv FROM agvworklist
WAIT

END

IF fromtob( agvberthnb OF tmagv,sbnr OF tmagv)=0
bye:
IF TMAGV IS ACTIVE
CANCEL TMAGV
END
ACTIVATE tmagv FROM toparking IN agvmod IF tmagv IS NOT ACTIVE
REMOVE tmagv FROM agvworklist

WAIT
END

@--agv has just received a shufflecontainer in own stack, and is about to @
@-- deliver it after receiving location of the next stack -->

IF loadedagv OF tmagv = TRUE & (CEIL((y OF tmagv)/10)= (x OF tmagv))
JUMP TO yardshuffeleallocation
JUMP TO agv_stack_stack
IF firstshuffle OF tmagv = TRUE
firstshuffle OF tmagv = FALSE
END
JUMP TO y_creation @nieuw
@JOIN tmagv TO assignedqueue OF agvcrane OF tmagv @nieuw
IF TMAGV IS ACTIVE
CANCEL TMAGV
END
ACTIVATE tmagv FROM shuffling IN agvmod IF tmagv IS NOT ACTIVE
REMOVE tmagv FROM agvworklist
WAIT
END

@--agv has just delivered the container to its new destination and wants--@
@-- a container to bring back to its own stack (if there is any) -->
IF loadedavg OF tmagv = FALSE & (CEIL((y OF tmagv)/10) ≠ (x OF tmagv))

    IF fromtob(sbnr OF tmagv,agvberthnb OF tmagv)>0
        fromtob(sbnr OF tmagv,agvberthnb OF tmagv)--fromtob--
        [sbnr OF tmagv,agvberthnb OF tmagv]-1
        yyy OF tmagv--y OF tmagv
        xxx OF tmagv--FLOOR((yyy OF tmagv-1)/20)
        zzz OF tmagv--FLOOR((yyy OF tmagv-1)/20)
        yyy OF tmagv--(yyy OF tmagv)-(zzz OF tmagv)*20
        IF (yyy OF tmagv)MODULO(2)=0
            yyy OF tmagv = yyy OF tmagv -1
        END
    @IF (yyy OF tmagv)> 10
        @yyy OF THIS agv-- yyy OF tmagv -10
    @END

    yyy OF tmagv = yyy OF tmagv + 20*(xxx OF tmagv)
    tmcrane= FIRST crane IN cranelist WITH cranelumber = yyy OF tmagv
    JOIN tmagv TO assignedqueue OF tmcrane

    JOIN tmagv TO cranequeue OF tmcrane IN FRONT POSITION
    REMOVE tmagv FROM agworklist

WAIT

END

IF fromtob(sbnr OF tmagv,agvberthnb OF tmagv)=0
    IF fromtob(agvberthnb OF tmagv,CEIL((y OF tmagv)/60))=0
        JUMP TO yardshuffleallocation
        JUMP TO agv_stack_stack
        bye OF tmagv = TRUE
        IF TMAGV IS ACTIVE
            CANCEL TMAGV
        END
        ACTIVATE tmagv FROM shuffling IN agvmod IF tmagv IS NOT ACTIVE
        REMOVE tmagv FROM agworklist
    END

WAIT

END

fromtob( agvberthnb OF tmagv,CEIL((y OF tmagv)/60))--fromtob--
    [agvberthnb OF tmagv,CEIL((y OF tmagv)/60)]-1
    JUMP TO yardshuffleallocation
    JUMP TO agv_stack_stack
    IF TMAGV IS ACTIVE
    CANCEL TMAGV
END

ACTIVATE tmagv FROM shuffling IN agvmod IF tmagv IS NOT ACTIVE
REMOVE tmagv FROM agworklist

WAIT

END

IF loadedavg OF tmagv = TRUE & (CEIL((y OF tmagv)/10) ≠ (x OF tmagv))
IF fromtobj agvberthnb OF tmagv,CEIL((y OF tmagv)/60)] = 0
  JUMP TO yardshuffleallocation
  JUMP TO agv_stack_stack
  bye OF tmagv ← TRUE
  IF TMAGV IS ACTIVE
    CANCEL TMAGV
  END
  ACTIVATE tmagv FROM shuffling IN agvmod IF tmagv IS NOT ACTIVE
  REMOVE tmagv FROM agvworklist
WAIT
END
JUMP TO yardshuffleallocation
JUMP TO agv_stack_stack

IF TMAGV IS ACTIVE
  CANCEL TMAGV
END
ACTIVATE tmagv FROM shuffling IN agvmod IF tmagv IS NOT ACTIVE
REMOVE tmagv FROM agvworklist
WAIT
END

WAIT
END
WAIT
The crane does not do anything until an AGV is assigned or an AGV enters a cranequeue.

If the length of the cranequeue > 0 there is an AGV waiting under the crane Line 8 to 25

If aantalagv > 0 it means an AGV has dropped its container on a service point. The AGV is gone and the crane recognises that a container has to be transported into the warehouse.

Line 28 readyqueue is a set containing AGV's that have dropped a container onto a service point. This does not mean the AGV still has to be in the warehouse.

`crane_cycletime` is a macro which creates a cycle time for the crane.

65 seconds together with the 15 seconds Line 31 are the working time for the crane to get the container into the right bay. This time is deterministic in contrast to Line 36 just one of the two can be used.

`stackcount` are the reserved service points in the stack.

`klaarcoun` is the counter that counts the containers that are placed on the service points by the crane, ready to be taken away by an AGV.

From Line 48 to 59 if stackcount >= 2 & klaarcoun >= 1 this means the crane has been anticipating and has already dropped a container onto a service point ready to be taken away by an AGV.
begin:

WAIT WHILE aantalagev OF THIS crane = 0 & LENGTH OF assignedqueue OF THIS crane = 0 & LENGTH OF craneladenqueue OF THIS crane = 0 & LENGTH OF cranelossenqueue OF THIS crane = 0

IF LENGTH OF cranequeue OF THIS crane > 0
    craneagev OF THIS crane ← FIRST AGV IN cranequeue OF THIS crane
    IF craneagev OF THIS crane BELONGS TO assignedqueue OF THIS crane
        assignedagev OF THIS crane ← craneagev OF THIS crane
        REMOVE assignedagev OF THIS crane FROM assignedqueue OF THIS crane
    END
    @GOTO halverwege
    END

IF loadedagev OF craneagev OF THIS crane = FALSE
    REMOVE craneagev OF THIS crane FROM cranequeue OF THIS crane
    JOIN craneagev OF THIS crane TO craneladenqueue OF THIS crane
    @GOTO begin
    END

IF loadedagev OF craneagev = TRUE
    REMOVE craneagev OF THIS crane FROM cranequeue OF THIS crane
    JOIN craneagev OF THIS crane TO cranelossenqueue OF THIS crane
    @GOTO begin
    END

IF aantalagev OF THIS crane > 0
    craneagev OF THIS crane ← FIRST OF readyqueue OF THIS crane
    REMOVE craneagev OF THIS crane FROM readyqueue OF THIS crane
    aantalagev OF THIS crane ← aantalagev OF THIS crane - 1
    WORK 15 SECONDS
    @cranelastagev OF THIS crane ← FIRST stack IN stacklist WITH stacknumber = yy
    cranelastagev OF THIS crane ← FIRST stack IN stacklist WITH stacknumber = zz
    stackcount OF cranelastagev OF THIS crane ← stackcount OF cranelaststack OF THIS crane
    remove craneagev FROM wantasqueue OF cranelaststack OF THIS crane IF craneagev OF cranelaststack OF THIS crane ≠ craneagev OF cranelastagev OF THIS crane
    @JUMP TO CRANE_CYCLETIME @HOREN BIJ HET EXPERIMENT
    @JUMP TO crane_outportations
    @WORK (F OF THIS CRANE -15)SECONDS @HOREN BIJ HET EXPERIMENT
    WORK 65 SECONDS
    @GOTO begin
    END

IF LENGTH OF cranelossenqueue OF THIS crane > 0
    cranelagev OF THIS crane ← FIRST agv IN cranelossenqueue OF THIS crane
    cranelagev OF THIS crane ← agvstack OF craneagev OF THIS crane
    REMOVE cranelagev FROM wantasqueue OF cranelaststack OF THIS crane IF cranelagev BELONGS TO wantasqueue OF cranelaststack OF THIS crane
    IF shuffleagev OF cranelagev = TRUE
        IF stackcount OF cranelaststack OF THIS crane ≥ 2 & klaarcount OF THIS crane ≥ 2
            REMOVE cranelagev FROM cranelossenqueue OF THIS crane
            stackcount OF cranelaststack ← stackcount OF cranelaststack - 1
            klaarcount OF THIS crane ← klaarcount OF THIS crane - 1
            @JOIN cranelagev TO readyqueue OF THIS crane
            @WAIT 15 SECONDS
            JOIN cranelagev OF THIS crane TO agvwantlist
            @ACTIVATE craneagev FROM begin IN agvmoe
        @GOTO final
        END
    END
    END

GOTO volleagev
If length of craneladenqueue > 0 line 64 to 82 it means there is an empty in the stack that wants a container if stackcount >= 2 & klaarcoun >= 1 there is a container ready for the rgv if not goto line 81

Assignedqueue is not empty, there is a rgv assigned line 84 to 126

If it is a shuffle rgv line 87 to 106

line 88 to 99 the crane gets a container for the assigned rgv and puts it on a service point this can be done because there is room at the service points

klaarcoun +1 this means there is a container ready to be taken away

Line 100 to 105 there is no service point ready for the crane to put a container on. In this case the crane cannot do anything yet for the assigned rgv

line 107 the rgv is assigned but is not a shuffle rgv, this means it has to be a rgv that wants a container to load the vessel

line 110 to 119 the crane gets a container from the warehouse and puts it on a service point this can be done because there is room at the service points

The crane can not yet get a container because the service points are taken there is nothing the crane can do at the moment for this rgv
63 END
64 IF LENGTH OF craneladenqueue OF THIS crane > 0
65 craneagv OF THIS crane ← FIRST OF craneladenqueue OF THIS crane
66 cranestack OF THIS crane ← avystack OF craneagv OF THIS crane
67
68 REMOVE craneagv FROM wantasqueue OF cranestack OF THIS crane IF craneagv →
69 BELONGS TO wantasqueue OF cranestack OF THIS crane
70 IF shuffleagv OF craneagv = TRUE
71 IF stackcount OF cranestack ≥ 2 & klaarcount OF THIS crane ≥ 1
72 REMOVE craneagv FROM craneladenqueue OF THIS crane
73 stackcount OF cranestack ← stackcount OF cranestack - 2
74 klaarcount OF THIS crane ← klaarcount OF THIS crane - 1
75 JOIN craneagv OF THIS crane TO agvworklist
76 ACTIVATE craneagv FROM begin IN agvmod
77 loadedagv OF craneagv ← TRUE
78 frombto(b snr OF craneagv, agvberthnb OF craneagv) ← frombto(b snr OF c
79 GOTO final
80 END
81 END
82 GOTO legeagv
83 END
84 IF assignedqueue OF THIS crane IS NOT EMPTY
85 assignedagv OF THIS crane ← FIRST agv IN assignedqueue OF THIS crane
86 halverwege:
87 IF shuffleagv OF assignedagv OF THIS crane = TRUE
88 cranestack OF THIS crane ← FIRST stack IN stacklist WITH stacknumber = y OF
89 REMOVE craneagv FROM wantasqueue OF cranestack OF THIS crane
90 IF stackcount OF cranestack OF THIS crane ≤ 1
91 stackcount OF cranestack OF THIS crane ← stackcount OF cranestack OF T.
92 REMOVE assignedagv OF THIS crane FROM assignedqueue OF THIS crane
93 @JUMP TO CRANE_CYCLETIME @HOREN BIJ HET EXPERIMENT
94 @JUMP TO crane_outportations
95 @WORK (F OF THIS CRANE )SECONDS @HOREN BIJ HET EXPERIMENT
96 WORK 80 SECONDS
97 klaarcount OF THIS crane ← klaarcount OF THIS crane + 1
98 GOTO begin
99 END
100 IF stackcount OF cranestack OF THIS crane > 1
101 REMOVE assignedagv FROM assignedqueue OF THIS crane
102 JOIN assignedagv OF THIS crane TO wantasqueue OF cranestack OF THIS c
103 ACTIVATE cranestack OF THIS crane FROM begin IN stackmod IF cranesta-
104 GOTO begin
105 END
106 END
107 IF shuffleagv OF assignedagv OF THIS crane = FALSE
108 cranestack OF THIS crane ← FIRST stack IN stacklist WITH stacknumber = y OF
109 REMOVE craneagv FROM wantasqueue OF cranestack OF THIS crane IF craneagv :
110 wantasqueue OF cranestack OF THIS crane
111 IF stackcount OF cranestack OF THIS crane ≤ 2
112 stackcount OF cranestack OF THIS crane ← stackcount OF cranestack OF
113 REMOVE assignedagv OF THIS crane FROM assignedqueue OF THIS crane
114 @JUMP TO CRANE_CYCLETIME @HOREN BIJ HET EXPERIMENT
115 @JUMP TO crane_outportations
116 @WORK (F OF THIS CRANE )SECONDS @HOREN BIJ HET EXPERIMENT
117 WORK 80 SECONDS
118 klaarcount OF THIS crane ← KLAARCOUNT OF THIS CRANE + 1
119 GOTO begin
120 END
121 IF stackcount OF cranestack OF THIS crane > 2
122 REMOVE assignedagv OF THIS crane FROM assignedqueue OF THIS crane
123 JOIN assignedagv OF THIS crane TO wantasqueue OF cranestack OF THIS c
124 IF ASSIGNEDAGV BELONGS NOT TO WANTASQUEUE OF CRANESTACK OF THIS CRA
Loaded rgv arrives at the crane

line 129 to 160 the crane serves the loaded agv

empty rgv arrives at the crane
GOTO begin
END
END

vollea1:
cranegal OF THIS crane ← FIRST agv IN craneloadedqueue OF THIS crane
REMOVE craneagv FROM assignedqueue OF THIS crane IF craneagv BELONGS TO assignedqueue OF THIS crane
CANCEL craneagv OF THIS crane IF craneagv OF THIS crane IS ACTIVE
IF loadedagv OF craneagv OF THIS crane = TRUE
loadedagv OF craneagv ← FALSE
REMOVE craneagv FROM craneloadedqueue→
OF THIS crane IF craneagv BELONGS TO craneloadedqueue OF THIS crane
IF x OF craneagv = 0
JOIN craneagv TO agvworklist
ACTIVATE craneagv FROM begin IN agvmod
GOTO final
END
IF bye OF craneagv = TRUE
ACTIVATE craneagv FROM toparking IN agvmod
GOTO final
END
goizer:
@JUMP TO CRANE_CYCLETIME @HOREN BIJ HET EXPERIMENT
@JUMP TO crane_outportations
@WORK (F OF THIS CRANE )SECONDS @HOREN BIJ HET EXPERIMENT
WORK 80 SECONDS
JOIN craneagv TO agvworklist
ACTIVATE craneagv FROM begin IN agvmod
GOTO final
END
empty agv arrives at the crane @
leage1:
cranegal ← FIRST agv IN craneloadedqueue OF THIS crane
REMOVE craneagv FROM assignedqueue OF THIS crane IF craneagv BELONGS TO assignedqueue OF THIS crane
IF loadedagv OF craneagv = FALSE
IF bye OF craneagv = TRUE & shuffleagv OF craneagv = TRUE
ACTIVATE craneagv FROM toparking IN agvmod
REMOVE craneagv OF THIS crane FROM craneloadedqueue OF THIS crane
GOTO final
END
prima:
@JUMP TO CRANE_CYCLETIME @HOREN BIJ HET EXPERIMENT
@JUMP TO crane_outportations
@WORK (F OF THIS CRANE )SECONDS @HOREN BIJ HET EXPERIMENT
WORK 80 SECONDS
loadedagv OF craneagv OF THIS crane ← TRUE
@prima:
IF bye OF craneagv OF THIS crane = TRUE & shuffleagv OF craneagv = TRUE
ACTIVATE craneagv FROM toparking IN agvmod
REMOVE craneagv FROM craneloadedqueue OF THIS crane
GOTO final
END
IF loadedagv OF craneagv OF THIS crane = FALSE
    loadedagv OF craneagv OF THIS crane = TRUE
END

REMOVE craneagv OF THIS crane FROM craneladenqueue OF THIS crane
JOIN craneagv OF THIS crane TO agyworklist
ACTIVATE craneagv OF THIS crane FROM begin IN agvmod
yy OF craneagv OF THIS crane -y OF craneagv OF THIS crane
GOTO final

END
final:
GOTO begin

202
Appendix 28-III Macro's

MACROS
- CREATE_PLATFORM
- STATION
- SHORT_CUT
- STACK_CRANE
- AGV_TOPARKING
- RGV_UNLOAD
- RGV_UNLOAD_BACK
- AGV_STACK_BERTH
- AGV_BERTH_STACK
- AGV_STACK_STACK
- CREATE_STACK
- SUPER_SHORTCUT
- UNLOADYARDLOCATION
- LOADYARDLOCATION
- CONVERSATION
- PRECLAIMNODES
- AGV_UNEASY_B_S
- YARDSHUFFLELOCATION
- Y_CREATION
- CREATE_OUTPORTATION
- CRANE_CYCLETIME
- CRANE_OUTPORTATIONS
- CREATING_SECTIONS
- CREATE_PORTOUT
- PORTAINERTIMES
Each portainer location has two platforms these are the actual lanes under the portainer
A platform consists of two sections a way in and a way out

n_in is a reference to a node, node_in the entrance of the platform

pl_x & pl_y are the x and y co-ordinates of the platform

Platforms is a set containing all the platforms
1 THIS platform ← NEW platform CALLED char
2 way_in ← NEW section
3 way_out ← NEW section
4 row ← NEW SET
5 n_in ← READ FROM hconfig
6 n_out ← READ FROM hconfig
7 THIS node ← nodes[n_in]
8 outs[2] ← way_in
9 side_track[1] ← station
10 pl_x ← READ FROM hconfig
11 pl_y ← READ FROM hconfig
12 end_point OF way_in ← THIS platform
13 s_length OF way_in ← distance_to(pl_x, pl_y)
14 THIS node ← nodes[n_out]
15 end_point OF way_out ← THIS node
16 s_length OF way_out ← distance_to(pl_x, pl_y)
17 interval_check ← NEW POINTSTREAM
18 JOIN THIS platform TO platforms
19 RETURN
This is the time the rgv needs slow down and turn is wheels while it is on the node line 6

Line 5 & 6 This is the time the rgv needs to get of the main track and pass the entrance
7 node of the way_in
8 line 8 to 15 decide which exit to take. At the node there are two exits, because there
9 are two way_ins for two portainerlocations. The right exit is decided by the x of the rgv
10 x is an attribute of the rgv referring to a portainerlocation
11
12
13
14
15
16 exit_time is a specific time for a rgv to pull up and leave the node
17
18 At the moment the rgv has left the node it unclaims the node
19 s_length is the section length
20 this is the section way_in of the station
21
22 In line 22 to 33 the entrance node is given to the rgv it will need this node later when
23 after the portainer it want to continue its journey, the entrance node is the end point of
24 way out of the portainer there are two different entrance nodes to be given depending
25 on the x of the rgv
26
27
28
29
30
31
32
33
34
35
36 In line 35 and 36 the rgv has eventually reached the portainer and enters the
37 portlocqueue waiting to be served
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
@ a trolley enters a portainer to be @
@ handled and to obtain a next task @
--------------------------------------@

WORK (nodedistance1 OF cur_node)/(nodespeed OF cur_node)
WAIT 4 SECONDS

8 IF (x OF THIS agv) MODULO(2) # 1
  cur_sec - outs[3] OF cur_node
9 END

11 IF (x OF THIS agv) MODULO(2) = 1
  cur_sec - outs[2] OF cur_node
12 END

15 WORK exittime OF cur_node @optrekken vanuit stand
16 claimed OF cur_node = claimed OF cur_node - 1
18 WORK (s_length OF cur_sec/low_speed)+ 2*(low_speed/acc)
20
21 IF (x OF THIS agv) MODULO(2) # 1
22 cur_sec - outs[3] OF cur_node
23 cur_node-end_point OF cur_sec
24 JOIN cur_node TO entrance OF THIS agv IF entrance OF THIS agv IS EMPTY
25 END

28 IF (x OF THIS agv) MODULO(2) = 1
29 stop = end_point OF cur_sec
30 cur_sec-way_out OF stop
31 cur_node-end_point OF cur_sec
32 JOIN cur_node TO entrance OF THIS agv IF entrance OF THIS agv IS EMPTY
33 END

35 agvportloc-FIRST portloc IN portlocations WITH portlocnumber = x OF THIS agv
36 ENTER portlocqueue OF agvportloc
37
38
39
40
41 WAIT
42
43
44 RETURN
45
46
In line 6 to 186 a rgv is at a node with two outs, outs[1] refers to the main track and outs[2] refers to the stack the rgv has to go to the stack

The rgv has to slow down in line 13 and turn its wheels line 14

stack row a set containing all the rgv's that are in the stack at that moment
rgv pulls up
rgv declaims the node
Rgv drives into the stack

Line 30 to 42 keeps count of the stack

the rgv has to wait 15 seconds this is for the shuttle to drive onto the rgv
If line 32 counts this is a rgv unloading a vessel the rgv has dropped a container which eventually has to go to another berth to be taken away.

If line 35 counts this rgv is shuffling and has just delivered a container to the right stack
inplacecounter counts the containers that are in the right stack

If line 39 counts this rgv's shuffling and has just delivered a container to the right stack but not at the rgv's own berth.

In line 47 to 66 a loaded rgv enters a stack this rgv helps unloading a vessel

If stackcount > 3 all the service points are reserved
the rgv has to wait in the craneossequene

If stackcount < 3 the rgv gets a service point to drop the container

The rgv drops its container at a service point aantalagv +1 this takes time but the 15 seconds are already deducted in line 31

The rgv is ready and wants a new task from the trafficmaster "enters the agvworklist"
SHORT_CUT - page 1

@ The AGV arrives at an exit node. This exit node can be a @
@ regular exit node or an exception exit node. @

@ The AGV arrives at a stack-entrance exception node and identifies the node with @
@ 3 exit possibilities, outs[1], outs[2], and decides to enter the stack @

IF cur_node IS nodes[30*CEIL(y/20)+200-(y-11)MODULO(10)] & route OF THIS agv IS EM

WORK (nodedistance1 OF cur_node)/(nodespeed OF cur_node)
WORK 4 SECONDS @wieldraaiings boete?
agvstack-FIRST stack IN stacklist WITH stacknumber = y OF THIS agv
beeninstack OF THIS agv-TRUE
ENTER stack_row OF agvstack OF THIS agv

WORK exittime OF cur_node @optrekken tot in het stack

cur_sec = outs[3] OF cur_node

WORK ((s_length OF cur_sec)/(2*low_speed)) + 2*(low_speed/acc)

claimed OF cur_node = claimed OF cur_node - 1
agvstack-FIRST stack IN stacklist WITH stacknumber = y OF THIS agv

jump to y creation

agvcrate OF THIS agv- FIRST crane IN cranelist WITH cranenumber = yyy OF THIS
WAIT 15 SECONDS
IF shuffleag OF THIS agv ≠ TRUE & loadedagy OF THIS agv = TRUE
fromtob[t agvberthnb OF THIS agv, sbnr OF THIS agv] - fromtob[t agvberthnb OF
END
IF shuffleag OF THIS agv = TRUE & (sbnr OF THIS agv =0) & loadedagv OF THIS
inplacecounter[agvberthnb OF THIS agv] + inplacecounter[agvberthnb OF THIS
END
IF shuffleag OF THIS agv = TRUE & (sbnr OF THIS agv ≠0) & loadedagv OF THIS &
inplacecounter[sbnr OF THIS agv ] + inplacecounter[sbnr OF THIS agv] +1
END

REMOVE THIS agv FROM assignedqueue OF agvcrate IF THIS agv BELONGS TO assigne-
agvstack-FIRST stack IN stacklist WITH stacknumber = y OF THIS agv

REMOVE THIS agv FROM wantasques OF agvstack IF THIS agv BELONGS TO wantasque-

IF loadedagv OF THIS agv =TRUE & shuffleagv OF THIS agv = FALSE
yard OF THIS agv = TRUE
IF stackcount OF agvstack ≥ 3
ENTER cranelossenqueue OF agvcrate OF THIS agv
WAIT

IF stackcount OF agvstack < 3
stackcount OF agvstack = stackcount OF agvstack + 1
antalagy OF agvcrate = antalagy OF agvcrate +1
LEAVE agvworklist IF THIS agv BELONGS TO agvworklist
loadedagv OF THIS agv-FALSE
ENTER agvworklist
yY OF THIS agv = y OF THIS agv
rr OF THIS agv = y OF THIS agv
ENTER readyqueue OF agvcrate
WAIT WHILE trafficmaster IS ACTIVE
If these conditions count the rgv is helping to load a vessel line 68 to 90

if stackcount >=1 & klaarcount >=1 there is a container waiting at a service point to
be taken away by a rgv this happens in line 71 to 81
line 72 to 76 the properties of the stack and the rgv are adjusted and the rgv enters the
rgvworklist to attain a next assignment

in line 82 to 89 there is no container ready for the rgv, it has to wait in the
craneladenqueue

Line 94 to 181 a shuffle rgv enters the stack

In line 97 to 118 there is a container waiting for the rgv

If the rgv is not loaded it will not need a service point to drop its container so this
service point can be used for other purposes

If the condition in line 109 counts it means a rgv will deliver a container but there is no
shuffle container to take with him, it will leave the stack empty
antal +1 means a container is dropped on a service point

In line 119 to 134 a rgv enters its own stack
Own stack is a stack behind the portainer the rgv was working for in the beginning
In line 120 to 125 the rgv enters empty

Nietsterug is line 168 in this line the rgv enters the agvworklist to ask the trafficmaster
if he has to get another container from this stack.
ACTIVATE trafficmaster FROM begin IN trafficmastmod IF trafficmaster IS N
WAIT
END
END

IF loadedagv OF THIS agv = FALSE & shuffleagv OF THIS agv =FALSE

yard OF THIS agv ← TRUE
agvstack-FIRST stack IN stacklist WITH stacknumber = y OF THIS agv
IF stackcount OF agvstack ≥ 1 & klaarcount OF agvcrane OF THIS agv ≥ 1
  klaarcount OF agvcrane-klaarcount OF agvcrane -1
  stackcount OF agvstack ← stackcount OF agvstack -1
  loadedagv OF THIS agv ← TRUE
  yard OF THIS agv ← TRUE
  yy OF THIS agv ← y OF THIS agv
  ENTER agvworklist
  WAIT WHILE trafficmaster IS ACTIVE
  ACTIVATE trafficmaster FROM begin IN trafficmastmod IF trafficmaster IS N
  WAIT
END
IF stackcount OF agvstack =0
  ENTER craneladenqueue OP agvcrane OF THIS agv
  WAIT
END
IF klaarcount OF agvcrane =0
  ENTER craneladenqueue OP agvcrane OF THIS agv
  WAIT
END
IF shuffleagv OF THIS agv =TRUE

yard OF THIS agv ← TRUE
agvstack-FIRST stack IN stacklist WITH stacknumber = y OF THIS agv
IF stackcount OF agvstack ≥ 2 & KLAARCOUNT OF AGVCRANE OF THIS agv≥1
  stackcount OF agvstack ← stackcount OF agvstack -1
  IF agvberthnb OF THIS agv = (CEIL(stacknumber OF agvstack OF THIS agv /60
  & sbnr OF THIS agv # 0
  IF loadedagv OF THIS agv = FALSE & fromtobf agvberthnb OF THIS agv,sb:
    stackcount OF agvstack ← stackcount OF agvstack -1
    GOTO fromhere
END
IF loadedagv OF THIS agv = FALSE & fromtobf agvberthnb OF THIS agv,sb:
  stackcount OF agvstack ← stackcount OF agvstack -1
  GOTO leegin
END
IF fromtobf agvberthnb OF THIS agv,sbnr OF THIS agv)≤0
  JOIN THIS agv TO readyqueue OF agvcrane
  aantalagv OF agvcrane OF THIS agv ← aantalagv OF agvcrane OF THIS
  fromhere:
  rr OF THIS agv← y OF THIS agv
  yy OF THIS agv ← y OF THIS agv
  loadedagv OF THIS agv ← FALSE
  GOTO nietsterug
END
IF agvberthnb OF THIS agv = (CEIL(stacknumber OF agvstack OF THIS agv /60
  IF loadedagv OF THIS agv = FALSE
  stackcount OF agvstack ← stackcount OF agvstack -1
  rr OF THIS agv← y OF THIS agv
  yy OF THIS agv ← y OF THIS agv
  GOTO nietsterug
line 126 to 134 if loaded is true the container is dropped on a service point

132 goto nietsterug line 167 in this line the rgv enters the agvworklist to ask the trafficmaster
133 if he has to get another container from this stack.

136 In line 136 to 155 the shuffle agv enters a stack but not its own stack this is the same as line
137 100 to 132

155 The moment the rgy passes line 156 it means it is about to receive a container from the stack
157 These are the 20 seconds in line 156. The shuttle puts the container on the rgy

158
159
160
161
162
163
164
165
166
167

168 the rgy enters the agvworklist, requesting the agvmaster for work

169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
END
126 IF loadedagv OF THIS agv = TRUE
127     JOIN THIS agv TO readyqueue OF agvcrane
128     aantalagv OF agvcrane OF THIS agv ← aantalagv OF agvcrane OF THIS
129     rr OF THIS agv ← y OF THIS agv
130     yy OF THIS agv ← y OF THIS agv
131     loadedagv OF THIS agv ← FALSE
132     GOTO nietsterug
133 END
134
135
136 IF agvberthnb OF THIS agv ≠ (CEIL(stacknumber OF agvstack OF THIS agv /60
137     IF loadedagv OF THIS agv = FALSE & fromtob[sbnr OF THIS agv,agvberthb:
138         stackcount OF agvstack ← stackcount OF agvstack -1
139         GOTO fromthere
140     END
141     IF loadedagv OF THIS agv = FALSE & fromtob[sbnr OF THIS agv,agvberthb:
142         stackcount OF agvstack ← stackcount OF agvstack -1
143         GOTO leegin
144     END
145
146 IF fromtob[sbnr OF THIS agv,agvberthnb OF THIS agv] ≤0
147     JOIN THIS agv TO readyqueue OF agvcrane
148     aantalagv OF agvcrane OF THIS agv ← aantalagv OF agvcrane OF THIS
149     fromthere:
150     rr OF THIS agv ← y OF THIS agv
151     yy OF THIS agv ← y OF THIS agv
152     loadedagv OF THIS agv ← FALSE
153     GOTO nietsterug
154 END
155
156 WAIT 20 SECONDS
157 fromtob[sbnr OF THIS agv, agvberthnb OF THIS agv] ← fromtob[sbnr OF THI:
158 @ nu moet hij al de juiste eigenschappen meekrijgen
159 rr OF THIS agv ← y OF THIS agv
160 JOIN THIS agv TO readyqueue OF agvcrane
161 aantalagv OF agvcrane OF THIS agv ← aantalagv OF agvcrane OF THIS agv +1
162 leegin:
163 klaarcount OF agvcrane OF THIS agv ← klaarcount OF agvcrane OF THIS agv -1
164 loadedagv OF THIS agv ← TRUE
165 yy OF THIS agv ← y OF THIS agv
166 nietsterug:
167 ENTER agworklist
168 WAIT WHILE trafficmaster IS ACTIVE
169 ACTIVATE trafficmaster FROM begin IN trafficmastmod IF trafficmaster IS N:
170 WAIT
171 END
172 IF stackcount OF agvstack OF THIS agv≤1
173     IF loadedagv OF THIS agv = FALSE & stackcount OF agvstack OF THIS agv =1
174         klaarcount OF agvcrane ← klaarcount OF agvcrane -1
175     GOTO leegin
176 END
177 ENTER cranequeue OF agvcrane
178 WAIT
179 END
180 END
181 ENTER cranequeue OF agvcrane
182 WAIT
183 RETURN
184 END
In line 187 to 225 the rgv arrives at a shortcut exception behind a portainer and chooses to go right (examples of these nodes are node 250, 280, 310....)

if the condition in line 191 counts it means the rgv can choose between to portainerlocations and chooses to take the left one as a result of his x value, the second condition line 192 is the slowing down of the rgv at the exit node

The 4 seconds are the rotation of the wheels

exittime is a specific time for a rgv to pull up and leave the node

outs[3] refers to the left side

rgv drives the length of the shortcut

Line 203 and 204 are the nodes that have to be unclaimed for the rgv to enter the main track again

in line 206 to 214 the rgv waits [208] until the nodes are not claimed the moment they are not claimed the rgv starts driving [211]

the rgv activates claim 2 in the nodemod. This is so the node can declaim itself normally this is done by the rgv but this is one of the exceptions

4 seconds is the discrotation of the rgv

In line 225 to 261 the same happens as in line 187 to 224 except that the rgv takes the left exit
@AGV arrives at a shortcut exception behind the portainer
@ coming from the stack and chooses to go left

IF cur_node IS nodes[181+(CEIL((x+0.5)/2))*30] & (x OF THIS agv) MODULO(2) #1
  WORK (nodedistance1 OF cur_node)/(nodespeed OF cur_node)  @Extra remboete on
  WORK 4 SECONDS @wieldraings boete!?  @tot stilstand te kom
  WORK exittime OF cur_node
  cur_sec ← outs[3] OF cur_node
  claimed OF cur_node ← claimed OF cur_node - 1
  cur_node ← end_point OF cur_sec
  WORK MAX(0,(((s_length OF cur_sec)/(low_speed)) - 2*(exittime OF cur_node)))
  claim1 ← FIRST node IN node_list WITH node_number = (node_number OF cur_node)
  claim2 ← FIRST node IN node_list WITH node_number = (node_number OF cur_node)
  IF claimed OF cur_node > 0 | claimed OF claim1 > 0 | claimed OF claim2 > 0
    WORK 2*(exittime OF cur_node)  @remmen(*extra)
    WAIT WHILE claimed OF cur_node > 0 | claimed OF claim1 > 0 | claimed OF claim2 > 0
    ACTIVATE claim2 FROM special IN nodemod
    claimed OF cur_node ← claimed OF cur_node + 1
    WORK 2*(exittime OF cur_node)  @optrekken(*2)
    WORK 4 SECONDS
    GOTO better
END

claimed OF cur_node ← claimed OF cur_node + 1
WORK exittime OF cur_node
WORK 4 SECONDS
WORK (nodedistance1 OF cur_node)/(nodespeed OF cur_node)

GOTO better
END

@AGV arrives at a shortcut exception behind the portainer
@ coming from the stack and chooses to go right

IF cur_node IS nodes[181+(CEIL((x+0.5)/2))*30] & (x OF THIS agv) MODULO(2) #0
  WORK (nodedistance1 OF cur_node)/(nodespeed OF cur_node)  @Extra remboete on
  WORK 4 SECONDS @wieldraings boete!?  @tot stilstand te kom
  WORK exittime OF cur_node
  cur_sec ← outs[2] OF cur_node
  claimed OF cur_node ← claimed OF cur_node - 1
  cur_node ← end_point OF cur_sec
  WORK MAX(0,(((s_length OF cur_sec)/(low_speed)) - 2*(exittime OF cur_node)))
  claim1 ← FIRST node IN node_list WITH node_number = (node_number OF cur_node)
  claim2 ← FIRST node IN node_list WITH node_number = (node_number OF cur_node)
  IF claimed OF cur_node > 0 | claimed OF claim1 > 0 | claimed OF claim2 > 0
    WORK 2*(exittime OF cur_node)  @remmen(*extra)
    WAIT WHILE claimed OF cur_node > 0 | claimed OF claim1 > 0 | claimed OF claim2 > 0
    ACTIVATE claim2 FROM special IN nodemod
    claimed OF cur_node ← claimed OF cur_node + 1
rvg arrives at a regular exit node and the exit requires a 90 degree bend. **Line 262 to 313**

This is a bit different than when a rgv doesn't have to make a turn.
WORK 2*(exit time of cur_node) @optrekken (*2)
WORK 8 SECONDS
GOTO beter

CLAIMED OF cur_node claimed OF cur_node + 1
WORK exit time of cur_node
WORK 4 SECONDS
WORK (nodedistance1 OF cur_node)/(nodespeed OF cur_node)
GOTO beter

@ AGV arrives at a regular exit node only the exit node requires 90 degree bend
IF (node_number OF cur_node OF THIS agv) - (node_number OF lastnode OF THIS agv) =
  IF (((node_number OF cur_node) - 200) MODULO (30) = 0) || (node_number OF cur_node
  & (node_number OF cur_node) < 484
  GOTO skip
END
WORK (nodedistance1 OF cur_node)/(nodespeed OF cur_node) @Extra remboete @
   @tot stilstand te kom
WORK 4 SECONDS @weldraarios boete!!
WORK exit time of cur_node
cur_sec = outs[2] OF cur_node
claimed OF cur_node = claimed OF cur_node - 1
ENTER sec_row OF cur_sec
reference_node = FIRST node IN node_list WITH node_number = 202
WORK MAX(0, ((s_length OF cur_sec)/(nodespeed OF reference_node)) - 2*(exit time OF
cur_node - end_point OF cur_sec
IF claimed OF cur_node > 0
@ hier gaat het gebeuren
  IF LENGTH OF route OF THIS agv = 1
n-node_number OF exit OF THIS agv
exit = FIRST node IN route OF THIS agv
n-node_number OF cur_node OF THIS agv
  IF node_number OF cur_node OF THIS agv = node_number OF exit OF THIS agv
  \& (n = 218 | n = 248 | n = 278 | n = 308 | n = 338 | n = 368 | n = 398 | n = 428 | n = 4
  LEAVE sec_row OF cur_sec
  cur_node OF THIS agv = FIRST node IN route OF THIS agv
  REMOVE FIRST node IN route OF THIS agv FROM route OF THIS agv
  weird OF THIS agv = TRUE
RETURN
END

WORK 2*(exit time OF cur_node) @remmen (*1extra)
WAIT WHILE claimed OF cur_node > 0
claimed OF cur_node claimed OF cur_node + 1
WORK 2*(exit time OF cur_node) @optrekken (*2)
LEAVE sec_row OF cur_sec
GOTO beter

CLAIMED OF cur_node claimed OF cur_node + 1
WORK exit time OF cur_node
LEAVE sec_row OF cur_sec
The last possible option is a RGV takes a shortcut but doesn't have to take a turn.
GOTO beter

END

skip:

@ The only possibility that remains is that the agv exits without 90 dgr. turn

reference_node ← FIRST node IN node_list WITH node_number = 202
WORK (nodedistance1 OF reference_node)/(nodespeed OF reference_node)
cur_sec ← outs[2] OF cur_node
claimed OF cur_node ← claimed OF cur_node - 1
ENTER sec_row OF cur_sec
WORK MAX(0,((s_length OF cur_sec/(nodespeed OF reference_node)) - 2*(nodedistance1 (cur_node ← end_point OF cur_sec
 IF claimed OF cur_node > 0

@ hier gaat het gebeuren
IF LENGTH OF route OF THIS agv = 1
n-node_number OF exit OF THIS agv
exit ← FIRST node IN route OF THIS agv
n-node_number OF cur_node OF THIS agv
IF node_number OF cur_node OF THIS agv = node_number OF exit OF THIS .
& (n = 218 | n=248 | n=278 | n=308 | n=338 | n=368 | n=398 | n=428 | n=4
LEAVE sec_row OF cur_sec
cur_node OF THIS agv ← FIRST node IN route OF THIS agv
REMOVE FIRST node IN route OF THIS agv FROM route OF THIS agv
weird OF THIS agv ← TRUE
RETURN
END

WORK 2*(exitetime OF reference_node) @remmen(*lextra)
WAIT WHILE claimed OF cur_node >0
claimed OF cur_node−−claimed OF cur_node + 1
WORK 2*(exitetime OF reference_node) @optrekken(*2)
LEAVE sec_row OF cur_sec
GOTO beter
END

claimed OF cur_node−−claimed OF cur_node + 1
WORK (nodedistance1 OF reference_node)/(nodespeed OF reference_node)

@--AGV has to make a 90 dgr. turn to get on the fastlane--@
exit ← FIRST node IN route OF THIS agv
IF cur_node IS NOT exit & ((node_number OF cur_node)<200 | (node_number OF cur_node)
WORK 4 SECONDS
WORK (nodedistance1 OF cur_node)/(nodespeed OF cur_node)
END

LEAVE sec_row OF cur_sec
beter:
exit ← FIRST node IN route OF THIS agv
REMOVE FIRST node IN route OF THIS agv FROM route OF THIS agv
373 final:
374 RETURN
375
376
377
378
The stacking crane macro works the same as line 1 to 180 of the short cut.
WORK (nodedistance1 OF cur_node)/(nodespeed OF cur_node)
7 WORK 8 SECONDS
8 agvstack-FIRST stack IN stacklist WITH stacknumber = y OF THIS agv
9 beeninstack OF THIS agv-TRUE
10 ENTER stack_row OF agvstack OF THIS agv
11
12 WORK exittime OF cur_node
13
14 cur_sec ← outs[2] OF cur_node
15
16 WORK (s_length OF cur_sec/low_speed) + 2*(low_speed/acc)  @!!!!
17 claimed OF cur_node ← claimed OF cur_node - 1
18 Jump to y creation
19
20 agvcane- FIRST crane IN cranelist WITH cranenumber = yyy OF THIS agv
21
22 WAIT 15 SECONDS
23 IF shuffleagv OF THIS agv # TRUE & loadedagv OF THIS agv = TRUE
24 FROMTBO[agvberthnb OF THIS agv, sbnr OF THIS agv]-FROMTBO[agvberthnb OF
25 END
26 IF shuffleagv OF THIS agv = TRUE & (sbnr OF THIS agv =0) & loadedagv OF THIS agv
27 inplacecounter[agvberthnb OF THIS agv]- inplacecounter[agvberthnb OF
28 END
29 IF shuffleagv OF THIS agv = TRUE & (sbnr OF THIS agv) #0 & loadedagv OF THIS agv
30 inplacecounter[sbnr OF THIS agv]- inplacecounter[sbnr OF THIS agv]+1
31 END
32
33 REMOVE THIS agv FROM assignedqueue OF agvcane OF THIS agv IF THIS agv BELONGS TO
34 agvstack OF THIS agv-FIRST stack IN stacklist WITH stacknumber = y OF THIS agv
35 REMOVE THIS agv FROM wantaasqueue OF agvstack OF THIS agv IF THIS agv BELONGS TO w.
36
37 IF loadedagv OF THIS agv =TRUE & shuffleagv OF THIS agv = FALSE
38 yard OF THIS agv ← TRUE
39 IF stackcount OF agvstack ≥ 3
40 ENTER cranelossenqueue OF agvcane OF THIS agv
41 WAIT
42 END
43 IF stackcount OF agvstack < 3
44 stackcount OF agvstack ← stackcount OF agvstack + 1
45 aantalagy OF agvcane ← aantalagy OF agvcane +1
46 LEAVE agvworklist IF THIS agv BELONGS TO agvworklist
47 loadedagv OF THIS agv=FALSE
48 yy OF THIS agv ← y OF THIS agv
49 ENTER agvworklist
50 rr OF THIS agv ← y OF THIS agv
51 ENTER readyqueue OF agvcane
52 WAIT WHILE trafficmaster IS ACTIVE
53 ACTIVATE trafficmaster FROM begin IN trafficmastmod IF trafficmaster IS N
54 WAIT
55 END
56 END
57
STACK_CRANE - page 2

63 IF loadedagv OF THIS agv = FALSE & shuffleagv OF THIS agv = FALSE
64   yard OF THIS agv ← TRUE
65
66  agvstack-FIRST stack IN stacklist WITH stacknumber = y OF THIS agv
67  IF stackcount OF agvstack ≥ 2 & klaarcoun OF agvcrane OF THIS agv ≥ 1
68       klaarcoun OF agvcrane-Klaarcoun OF agvcrane -1
69       stackcount OF agvstack ← stackcount OF agvstack -1
70       loadedagv OF THIS agv ← TRUE
71       yy OF THIS agv ← y OF THIS agv
72       ENTER agvworklist
73       WAIT
74
75  END
76
77  IF stackcount OF agvstack = 0
78     ENTER craneladenqueue OF agvcrane OF THIS agv
79     WAIT
80
81  END
82
83 END
84
85 IF shuffleagv OF THIS agv = TRUE
86   yard OF THIS agv ← TRUE
87
88  agvstack-FIRST stack IN stacklist WITH stacknumber = y OF THIS agv
89  IF stackcount OF agvstack ≥ 2 & klaarcoun OF AGVCRANE OF THIS agv ≥ 1
90       stackcount OF agvstack ← stackcount OF agvstack -1
91       IF agvberthnb OF THIS agv = (CEIL(stacknumber OF agvstack OF THIS agv / 60
92             IF loadedagv OF THIS agv = FALSE & fromtob[ agvberthnb OF THIS agv, sbn]
93             stackcount OF agvstack ← stackcount OF agvstack -1
94             GOTO fromhere
95
96  END
97
98  IF loadedagv OF THIS agv = FALSE & fromtob[ agvberthnb OF THIS agv, sbn]
99     stackcount OF agvstack ← stackcount OF agvstack -1
100    GOTO leegin
101
102 IF fromtob[ agvberthnb OF THIS agv, sbn] OF THIS agv] ≤ 0
103     JOIN THIS agv TO readyqueue OF agvcrane
104     aantalagv OF agvcrane OF THIS agv ← aantalagv OF agvcrane OF THIS
105     fromhere:
106     rr OF THIS agv ← y OF THIS agv
107     yy OF THIS agv ← y OF THIS agv
108     loadedagv OF THIS agv ← FALSE
109     GOTO nietsterug
110
111 END
112
113 IF agvberthnb OF THIS agv = (CEIL(stacknumber OF agvstack OF THIS agv / 60
114     & sbn] OF THIS agv = 0
115     IF loadedagv OF THIS agv = FALSE
116       stackcount OF agvstack ← stackcount OF agvstack -1
117       rr OF THIS agv ← y OF THIS agv
118       yy OF THIS agv ← y OF THIS agv
119       GOTO nietsterug
120
121 IF loadedagv OF THIS agv = TRUE
122     JOIN THIS agv TO readyqueue OF agvcrane
123     aantalagv OF agvcrane OF THIS agv ← aantalagv OF agvcrane OF THIS
124     rr OF THIS agv ← y OF THIS agv
125     yy OF THIS agv ← y OF THIS agv
126     loadedagv OF THIS agv ← FALSE
GOTO nietsterug
END
END

IF agvberthenb OF THIS agv ≠ CEIL(stacknumber OF agvstack OF THIS agv /60
  IF loadedagv OF THIS agv = FALSE & frommbtof{ sbnr OF THIS agv,agvberth:
    stackcount OF agvstack ← stackcount OF agvstack -1
    GOTO fromthere
  END
  IF loadedagv OF THIS agv = FALSE & frommbtof{ sbnr OF THIS agv,agvberth:
    stackcount OF agvstack ← stackcount OF agvstack -1
    GOTO leegin
  END
  IF frommbtof{ sbnr OF THIS agv,agvberthenb OF THIS agv} ≤ 0
    JOIN THIS agv TO readyqueue OF agvcrane
    aantalgv OF agvcrane OF THIS agv ← aantalgv OF agvcrane OF THIS
    fromthere:
    rr OF THIS agv ← y OF THIS agv
    yy OF THIS agv ← y OF THIS agv
    loadedagv OF THIS agv ← FALSE
    GOTO nietsterug
  END
END
WAIT 20 SECONDS
frommbtof{ sbnr OF THIS agv, agvberthenb OF THIS agv} ← frommbtof{ sbnr OF THI.
@ nu moet hij al de juiste eigenschappen meekrijgen
rr OF THIS agv ← y OF THIS agv
JOIN THIS agv TO readyqueue OF agvcrane
aantalgv OF agvcrane OF THIS agv ← aantalgv OF agvcrane OF THIS agv +1
leegin:
klarccount OF agvcrane OF THIS agv ← klarccount OF agvcrane OF THIS agv -1
loadedagv OF THIS agv ← TRUE
yy OF THIS agv ← y OF THIS agv
nietsterug:
ENTER agvworklist
WAIT WHILE trafficmaster IS ACTIVE
ACTIVATE trafficmaster FROM begin IN trafficmastmod IF trafficmaster IS Nk
WAIT
IF stackcount OF AGVSTACK OF THIS AGV = 1
  IF loadedagv OF THIS agv = FALSE & stackcount OF AGVSTACK OF THIS AGV = 1
    stackcount OF agvstack ← stackcount OF agvstack -1
    GOTO leegin
END
ENTER cranequeue OF agvcrane
WAIT
END
END
ENTER cranequeue OF agvcrane
WAIT
final:
RETURN
If the RGV was in a stack_row of a stack it is removed, normally it removes itself from a stack row.
If it starts driving after the stack. But in this case the RGV just disappears from the stack.

If the RGV was a shuffling for a vessel is now removed from that shuffle list.
@agv to parking@

1  tmx ← x OF THIS agv
2  tmy ← y OF THIS agv
3  agvstack-FIRST stack IN stacklist WITH stacknumber = tmy
4  LEAVE stack_row OF agvstack OF THIS agv IF THIS agv BELONGS TO stack_row-
5  OF agvstack OF THIS agv
6  JOIN THIS agv TO agvparking IF THIS agv BELONGS NOT TO agvparking

7  bye OF THIS agv ← FALSE
8  y OF THIS agv ← CEIL((tmx)/2)*20-18-
9  +((-1 + tmx)MODULO(2))*9
10
11  yard OF THIS agv ← TRUE
12  IF entrance OF THIS agv IS NOT EMPTY
13      REMOVE EACH node IN entrance OF THIS agv FROM entrance OF THIS agv
14  END
15  REMOVE THIS agv FROM agvberthshufflelist[ agvberthnb OF THIS agv] →
16  IF THIS agv BELONGS TO agvberthshufflelist[ agvberthnb OF THIS agv]
17
18  REMOVE THIS agv FROM agvworlist IF THIS agv BELONGS TO agvworlist
19  REMOVE THIS agv FROM activeagvlist IF THIS agv BELONGS TO activeagvlist
20
21  FOR i ← 1 TO 10
22      tmport ← FIRST portainer IN quayrail WITH portainernumber = i
23      REMOVE THIS agv FROM agvppactive OF tmport IF THIS agv BELONGS→
24          TO agvppactive OF tmport
25  END
26  WAIT
27
28  REMOVE THIS agv FROM agvppactive OF tmport IF THIS agv BELONGS TO agvppactive OR
29  WAIT
agv_unload is a macro that creates the route from the portainer to the right stack
there are two possibilities depending on the portainer location the rgv has just left

line 7 to 11 if the portainer location is odd this condition counts

line 12 to 15 if the portainer location is even this condition counts
begin:

tmx ← x OF tmagv

tmy ← y OF tmagv

IF (tmx) MODULO (2) = 1
    @cur_node
    goede_node-nodes[ 30*CEIL(tmy/20) + 200 - (tmy-1) MODULO (10) + (tmx-1) MODULO (2) ]
    JOIN goede_node TO route OF tmagv
END

IF (tmx) MODULO (2) = 0
    goede_node-nodes[ 30*CEIL(tmy/20) + 200 - (tmy-11) MODULO (10) ]
    JOIN goede_node TO route OF tmagv
END

RETURN
This macro creates the route from the stack back to the portainer. There are two possibilities:
1. In line 6 to 9, the entrance node is given, this is the end node of the stack.
2. Line 10 to 14 is for the odd numbered portainer location.
3. Line 15 to 19 is for the even numbered portainer location.
BEGIN:

yy OF tmagv = y OF tmagv

tmx = x OF tmagv

goede_stack = FIRST stack IN stacklist WITH stacknumber = yy OF tmagv

goede_section = way_out_stack OF goedese_stack @agvstack

goede_node = end_point OF goedese_section @agv_section

JOIN goede_node TO entrance OF tmagv IF entrance OF tmagv IS EMPTY

IF (tmx) MODULO(2) = 1

goede_node = nodes[212+CEIL((tmx-1)/2)*30]

JOIN goede_node TO route OF tmagv

END

IF (tmx) MODULO(2) = 0

goede_node = nodes[212+CEIL((tmx)/2)*30]

JOIN goede_node TO route OF tmagv

END

RETURN
`agv_stack_berth` is a macro that creates the route to get from a stack to a berth

First in line 5 to 8 the entrance node is given to the rgv

if the statement in line 11 counts the route has to be the same as expected in `agv_lossen`

A decision is made to let the odd rgv travel using the top route of the terminal line 22 to 187
and the even rgv use the bottom route of the terminal line 187 to the end

The rgv has to leave the main route connecting the stacks this rgv is odd so it has to leave the
stacks at the top line 25 and 26 create this node

the rgv has to go back (left) this is possible without first crossing another lane

just one node is required to tell the rgv where to leave this main track this is created in line 29

the rgv has to go foreword (right), first the rgv has to cross the driving lane that is directed to the
left this extra short cut is given in line 35 and 36 and line 37 creates the node where the rgv
should leave the main track again. Thereafter it has to cross the other lane again These nodes
could be (node260 line25, node116 line 35, node 85 if x is 4 line 37, node 114 line 39)

In line 39 the rgv has to cross a node again

This node in line 43 refers to the portainer location the rgv has to take, tmx refers to the portainer
location

If the condition in line 48 counts the rgv is located in a stack behind berth 1 or behind berth 3 and
has to go to the middle berth, berth 2.
The rgv has to take the top route this node is the exit of the stack area line 50 (for example if the
rgv is located in stack 143 this node will be 440)

Line 53 if the rgv is located behind berth 3 the rgv will have to go back from berth 3 to 2
This is explained in line 53 to 56 in line 54 the exit node is given.

If the rgv is located behind berth 1 the rgv will have to go foreword to berth 2 (right). It already has
the node from line 50 and 51 to exit the stack area but because it has to go to the right it first has
to cross a lane, this in contrast to the rgv that has to go left (line 53 to 56)

Line 60 and 61 is the crossing of the lane

Line 62 is the exit from the top main route
tmx = x OF tmagv
3 tmy = y OF tmagv
4
goede_stack = FIRST stack IN stacklist WITH stacknumber = yy OF tmagv
5
goede_section = way out stack OF goede_stack
6
goede_node = end point OF goede_section
7
JOIN goede_node TO entrance OF tmagv IF entrance OF tmagv IS EMPTY
8
9
10 IF easyload OF tmagv = TRUE
11
12 goede_node = nodes[212+CEIL((tmx-1)/2)*30]
13
JOIN goede_node TO route OF tmagv
14
15 GOTO later
16
17 END
18
19
20 begin:
21
22 IF (agvnnumber OF tmagv) MODULO(2)=1 | (AGVINNUMBER OF TMAGV) MODULO(2)=0
23 IF tmx = CEIL(tmy/10)
24
25 goede_node = nodes[CEIL(tmy/20)*30+200]
26
JOIN goede_node TO route OF tmagv
27
28 IF (tmx) MODULO(2)=1
29 goede_node = nodes[118-(CEIL(CEIL((tmx+0.5)/2)-1))*2]
30
JOIN goede_node TO route OF tmagv
31
32 END
33
34 IF (tmx) MODULO(2)=0
35 @de agv moet heen
36 @goede_node = nodes[119-CEIL(tmy/20)*2] @eerst eentje extra oversteken
37 goede_node = end point OF outs[2] OF goede_node
38
JOIN goede_node TO route OF tmagv
39
goede_node = nodes[CEIL(CEIL((tmx+0.5)/2))*2+79]
40
JOIN goede_node TO route OF tmagv
41
goede_node = end point OF outs[2] OF goede_node
42
JOIN goede_node TO route OF tmagv
43
44 END
45
46
47 IF (tmx ≥7 & tmx≤12) & (CEIL(tmy/60)=1|CEIL(tmy/60)=3)
48
goede_node = nodes[CEIL(tmy/20)*30+200]
49
JOIN goede_node TO route OF tmagv
50
51 IF tmx<(tmy/10) @de agv moet terug
52 goede_node = nodes[118-(CEIL((tmx + 0.5)/2)-1)*2]
53
JOIN goede_node TO route OF tmagv
54
55 END
56
57 IF tmx≥(tmy/10) @de agv moet heen
58
goede_node = end point OF outs[2] OF goede_node
59
JOIN goede_node TO route OF tmagv
60
goede_node = nodes[(CEIL((tmx+0.5)/2))*2+79]
63 **Line 64 and 65** the rgv has to cross the track again
64
65
66
67
68 **line 68** is the entrance of a portainerlocation
69
70
71
72
73 In line **73 to 112** if this condition counts it means the rgv has to travel back, to the left.
74 tmx/6 =1 this means the rgv has to go to berth 1 & tmy/60 =2 this indicates the rgv is located in
75 a stack behind berth 2
76
77 **line 77** this is the exit node out of the stack area
78
79
80 After is has taken the exit node it has to cross two lanes **line 80&81** and **line 83&84**
81
82
83
84
85
86 In line **86 to 89** the exit node of the top main route is given
87
88
89
90
91 In line 91 to 95 the rgv has to cross 2 lanes again
92
93
94
95
96
97 In line 97 the entrance of the portainerlocation is given
98
99
100
101
102
103
104 In line **104 to 106** a rgv has to go from a stack located behind berth 3 to a portainerlocation
105 at berth 2 goto tweede ring pakken refers to **line 75**
106
107
108
109 In line **109 to 119** If this condition counts it means rgv has to go foreword to the right
110
111 This conditions means a rgv is located in a stack behind berth 1 and has to go to a portainer
112 location behind berth 2
113 This is the same as line **75 to 99** therefore the rgv has to continue from tweede ring pakken **line 75**
114
115 This means a rgv is located in a stack behind berth 2 and has to go to a portainerlocation at berth
116 here also the rgv has to continue from **line 75**
117
118
119
120
121 **If the rgv number is even this means the rgv has to take the bottom route**
122
123 If this condition counts It means the rgv is located in a stack behind the berth it has to go to.
124 There are two possibilities depending on the portainerlocation ( the x of the rgv)
JOIN goede_node TO route OF tmagv

goede_node ← end_point OF outs[2] OF goede_node
JOIN goede_node TO route OF tmagv
END

goede_node ← nodes[212+CEIL((tmx-1)/2)*30]
JOIN goede_node TO route OF tmagv
GOTO later
END

IF CEIL(tmx/6)<CEIL(tmy/60)
    IF CEIL(tmx/6)=1 & CEIL(tmy/60)=2 @de tweede ring pakken Tweede_ring_pakken:
    goedie_node ← nodes[CEIL(tmy/20)*30+200]
JOIN goede_node TO route OF tmagv
END

goede_node ← end_point OF outs[2] OF goede_node
JOIN goede_node TO route OF tmagv @oversteken

goede_node ← end_point OF outs[2] OF goede_node
JOIN goede_node TO route OF tmagv @oversteken
    IF tmx<(tmy/10) @de agv moet terug
goede_node ← nodes[78-(CEIL((CEIL(tmx+0.5)/2)-1))*2]
JOIN goede_node TO route OF tmagv
END

END

JOIN goede_node TO route OF tmagv @oversteken

goede_node ← end_point OF outs[2] OF goede_node
JOIN goede_node TO route OF tmagv @oversteken

goede_node ← end_point OF outs[2] OF goede_node
JOIN goede_node TO route OF tmagv @oversteken

goede_node ← nodes[212+CEIL((tmx-1)/2)*30]
JOIN goede_node TO route OF tmagv
GOTO later

END

END

IF CEIL(tmx/6)>CEIL(tmy/60)
    IF CEIL(tmx/6)=2 & CEIL(tmy/60)=1 @tweede ring pakken Tweede_ring_pakken
GOTO Tweede_ring_pakken
END

END

IF (agvnumber OF tmagv) MODULO(2)=0 @AGV is even, dus onderlangs) GOTO later
IF tmx=CEIL(tmy/10) @achter de portainer in het stack
If \((\text{tmx}) \mod (2) = 1\) the portainer location is odd and the \text{rgv} has to go to the right (foreword).

For example a \text{rgv} is located in stack 44 it has to take exit 273 (line 130) than it has to take exit 506 (line 132).

If \((\text{tmx}) \mod (2) = 0\) the portainer location is even and the \text{rgv} has to go to the left (back).

Line 137 is the same as line 130.

Because the \text{rgv} has to go back it has to cross a lane (line 139).

This is the exit to the right portainer location.

The \text{rgv} has to cross the lane again.

This is the entrance of the portainer location (line 147).

In line 157 to 183 The \text{rgv} is located in a stack behind berth 1 or 3 and has to go to berth 2. This is the same as line 46 to 71 except here the \text{rgv}'s take the bottom route instead of the top route.

In line 185 the \text{rgv} is located in a stack behind berth 2 and has to go to berth 1. This is the same as line 75 except this time the \text{rgv} takes the bottom route instead of the top route.
@de crane heeft al een entrance node gegeven@
@dit is een andere dan bij de oneven agv's@

IF (tmx)MODULO(2)=1
  @de agv moet heen
  goede_node_nodes(213+30*CEIL(CEIL(((tmx/10)-1)/2)))
  JOIN goede_node TO route OF tnvag
  goede_node_nodes(500+ CEIL((CEIL(tmx)/10)/2))*2
  JOIN goede_node TO route OF tnvag
  END

IF (tmx)MODULO(2)=0
  @de agv moet terug
  goede_node_nodes(213+CEIL((tmx/10-1)/2)*30]
  JOIN goede_node TO route OF tnvag
  goede_node ← end_point OF outs[2] OF goede_node
  JOIN goede_node TO route OF tnvag @oversteken
  goede_node_nodes(539-(CEIL(CEIL(tmx))/2)*2]
  JOIN goede_node TO route OF tnvag
  goede_node ← end_point OF outs[2] OF goede_node
  JOIN goede_node TO route OF tnvag @oversteken
  END

  goede_node ← nodes(188+(CEIL(tmx/2))*30]
  JOIN goede_node TO route OF tnvag

GOTO later

END

@IF CEIL(tmx/6)=CEIL(tmy/60) @stack zit in dezelfde berth

IF (tmx ≥ 27 & tmx≤120) & (CEIL(tmx/60)=1|CEIL(tmy/60)=3)
  @AGV van 2 in berth op 2 gaat terug naar stack 1 of 3 via dichbij baan
  @de crane heeft al een entrance node gegeven@
  @dit is een andere dan bij de oneven agv's@

  goede_node_nodes(213+CEIL(((tmx/10)-1)/2)*30]
  JOIN goede_node TO route OF tnvag

  IF tnx>(tmx/10) @de agv moet heen
  goede_node_nodes(500+ CEIL((CEIL(tmx)/2))*2]
  JOIN goede_node TO route OF tnvag
  END

  IF tnx<(tmx/10)
    @de agv moet terug
    goede_node ← end_point OF outs[2] OF goede_node
    JOIN goede_node TO route OF tnvag @oversteken
    goede_node_nodes(539- CEIL(CEIL((tmx)/2))*2]
    JOIN goede_node TO route OF tnvag
    goede_node ← end_point OF outs[2] OF goede_node
    JOIN goede_node TO route OF tnvag @oversteken
    END

  goede_node ← nodes(188+(CEIL(tmx/2))*30]
  JOIN goede_node TO route OF tnvag

GOTO Later

END

IF CEIL(tmx/6)=1 & CEIL(tmy/60)=2 @tweede ring van onderen
Tweede_ring_van_onderen:
The rgv is located in a stack behind berth 3 and has to go to berth 2 here is referred to line 186

The rgv is located in a stack behind berth 1 and has to go to berth 2 here is referred to line 186

The rgv Located in a stack behind berth 2 and has to go to berth 3 here is referred to line 186
@de crane heeft al een entrance node gegeven
@dit is een andere dan bij de oneven agv's

goede_node-nodes[ 213+CEIL((tmy/10)-1)/2)*30]
JOIN goede_node TO route OF tmagv

GO goede_node - end_point OF outs[ 2] OF goede_node
JOIN goede_node TO route OF tmagv @ oversteken

GO goede_node - end_point OF outs[ 2] OF goede_node
JOIN goede_node TO route OF tmagv @ oversteken

IF tmx>(tmy/10) @de agv moet heen
GO goede_node-nodes[ 540+(CEIL(tmx/2))*2]
JOIN goede_node TO route OF tmagv
END

IF tmx<(tmy/10) @de agv moet terug
GO goede_node - end_point OF outs[ 2] OF goede_node
JOIN goede_node TO route OF tmagv @ oversteken

GO goede_node-nodes[ 579 - (CEIL(tmx/2))*2]
JOIN goede_node TO route OF tmagv

GO goede_node - end_point OF outs[ 2] OF goede_node
JOIN goede_node TO route OF tmagv @ oversteken

END

GO goede_node - end_point OF outs[ 2] OF goede_node
JOIN goede_node TO route OF tmagv @ oversteken

GO goede_node - end_point OF outs[ 2] OF goede_node
JOIN goede_node TO route OF tmagv @ oversteken

GO goede_node - nodes[ 188+(CEIL(tmx/2))*30]
JOIN goede_node TO route OF tmagv

GOTO later
END

IF CEIL(tmx/6)=2 & CEIL(tmy/60)=3
GOTO Tweede_ring_van_onderen
END

IF CEIL(tmx/6)=2 & CEIL(tmy/60)=1
GOTO Tweede_ring_van_onderen
END

IF CEIL(tmx/6)=3 & CEIL(tmy/60)=2
GOTO Tweede_ring_van_onderen
END

END
later:
RETURN
Agv_berth_stack is a macro that creates the route from a berth to the stack
The idea of this macro is the same as the macro agv_stack_berth off course the routes created are
different

Line 9 to 14
Firstshuffle is the transition for a rgv from the loading of a vessel to shuffling the first shuffle
container is retrieved from the stacks behind the portainerlocation the rgv was working
Easyload is an attribute of a rgv referring to the easy loading method of a vessel, this rgv retr
the load containers from the stacks behind the berth
In line 11 the node referred to is the entrance of a stack is joined to the route of the rgv
In line 17 to 185 the difference is made between a odd and a even rgv
Odd rgv's take the top route at the terminal when they want to get somewhere at the terminal
Even rgv's take the bottom route when driving at the terminal
If the condition in line 18 counts the rgv has to go from a portainerlocation to a stack located
behind this portainerlocation (Line 17 to 42)

If the condition in line 44 counts the rgv has to go from a portainerlocation at a berth to a stack
located behind the same berth Line 44 to 70
Line 46 this condition states the rgv is at berth 2 and has to go to stack located behind berth 1 or 3

This document contains natural text and does not require any visual representation or additional information.
AGV_BERTH_STACK - page 1

1 begin:
2 @
3 agvstack OF tmagv-FIRST stack IN stacklist WITH stacknumber = y OF tmagv
4 5 tmy-y OF tmagv
6 tmx-x OF tmagv
7 8
9 IF easyload OF tmagv = TRUE | firstshuffle OF tmagv = TRUE
10 11 goede_node ← nodes[200-(tmy-1)MODULO(10)+ CEIL(tmy/20)*30]
12 JOIN goede_node TO route OF tmagv
13 GOTO later
14 END
15 16
17 IF (agvnumber OF tmagv)MODULO(2)=1 @ agv is oneven dus bovenlangs
18 IF tmx=CEIL(tmy/10)
19 @achter de portainer in het stack
20 @de portainer heeft al een entrance node gegeven@
21 22 goede_node ← nodes[CEIL(tmx/2)*30+200]
23 JOIN goede_node TO route OF tmagv
24 25 IF (tmx)MODULO(2)=1
26 @de agv moet terug
27 goede_node ← nodes[118-((CEIL(tmx/10)+0.5)/2)-1]*2]
28 JOIN goede_node TO route OF tmagv
29 END
30 31 IF (tmx)MODULO(2)=0
32 @de agv moet heen
33 goede_node ← end_point OF outs[2] OF goede_node
34 JOIN goede_node TO route OF tmagv
35 goede_node ← nodes[(CEIL((CEIL(tmx/10)+0.5)/2))*2+79]
36 JOIN goede_node TO route OF tmagv
37 goede_node ← end_point OF outs[2] OF goede_node
38 JOIN goede_node TO route OF tmagv
39 END
40 41 goede_node-nodes[200+tmy+10*FLOOR(tmy/20.1)+20*((tmy-1)MODULO(2))]
42 JOIN goede_node TO route OF tmagv
43 GOTO later
44 END
45 46 @IF CEIL(tmx/6)=CEIL(tmx/60) @stack zit in dezelfde berth
47 48 IF (tmx ≥ 7 & tmx≤12) & (CEIL(tmx/60)=1|CEIL(tmx/60)=3)
49 @AGV van 2 in berth op 2 gaat terug naar stack 1 of 3 via dichbij baan
50 @de portainer heeft al een entrance node gegeven@
51 goede_node ← nodes[CEIL(tmx/2)*30+200]
52 JOIN goede_node TO route OF tmagv
53 54 IF tmy≥(tmy/10) @de agv moet terug
55 goede_node ← nodes[118-((CEIL(tmy/10)+0.5)/2)-1]*2]
56 JOIN goede_node TO route OF tmagv
57 END
58 59 IF tmy<(tmy/10) @de agv moet heen
60 goede_node ← end_point OF outs[2] OF goede_node
61 JOIN goede_node TO route OF tmagv
62 goede_node ← nodes[(CEIL((CEIL(tmx/10)+0.5)/2))*2+79]
63 JOIN goede_node TO route OF tmagv
In line 72 to 116 the rgv is located at a berth and has to go to a stack this stack is located at the right side of the portainerlocation. If the condition in Line 73 counts the rgv is at berth 1 and has to go to a stack located behind berth 2

Line 113 to 115 refers to a rgv at berth 2 that has to go to a stack located behind berth 3

If the condition in line 118 counts the rgv is a berth and has to go to a stack at the left side of this portainerlocation

Line 120 to 121 refer to a rgv at berth 2 that has to go to a portainerlocation tweede ring pakken in line 121 refers to line 74

If this condition counts the rgv is located at berth 3 and has to go to a portainerlocation behind berth 2 (line 124)
ga\node \leftarrow \text{end point of outs[2]}$ \text{of goede_node}$
45 \hspace{1em} \text{JOIN goede_node TO route of tmagv}
46 \text{END}

g\text{ode_node-nodes}[200+tmy+10^{*}\text{FLOOR}(tmy/20.1)+20^{*}(\text{tmy-1})\text{MODULO}(2))]
48 \hspace{1em} \text{JOIN goede_node TO route of tmagv}
49 \text{GOTO later}
50 \text{END}

51 \text{IF CEIL(tmx/6)<CEIL(tmy/60)}
52 \hspace{1em} \text{IF CEIL(tmx/6)=1 \& CEIL(tmy/60)=2 @de tweede ring pakken}
53 \hspace{1em} \text{Tweedee_ring_pakken:}
54 \hspace{1em} \text{de portainer heeft al een entrance node gegeven@}
55 \hspace{1em} \text{goede_node \leftarrow nodes[CEIL(tmx/2)*30+200]}
56 \hspace{1em} \text{JOIN goede_node TO route of tmagv}
57 \text{goede_node \leftarrow \text{end point of outs[2]} \text{of goede_node}}
58 \hspace{1em} \text{JOIN goede_node TO route of tmagv} \hspace{1em} @\text{oversteken}
59 \hspace{1em} \text{IF tmx>\text{CEIL(tmy/10)} @de agv moet terug}
60 \hspace{1em} \text{goede_node \leftarrow nodes[78-(\text{CEIL}((\text{CEIL(tmx/10)}+0.5)/2)-1))*2]}
61 \hspace{1em} \text{JOIN goede_node TO route of tmagv}
62 \text{END}

63 \text{IF tmx<\text{CEIL(tmx/10)} @de agv moet heen}
64 \hspace{1em} \text{goede_node \leftarrow \text{end point of outs[2]} \text{of goede_node}}
65 \hspace{1em} \text{JOIN goede_node TO route of tmagv}
66 \hspace{1em} \text{goede_node \leftarrow nodes[(\text{CEIL}((\text{CEIL(tmx/10)}+0.5)/2))*2+39]}
67 \hspace{1em} \text{JOIN goede_node TO route of tmagv}
68 \hspace{1em} \text{goede_node \leftarrow \text{end point of outs[2]} \text{of goede_node}}
69 \hspace{1em} \text{JOIN goede_node TO route of tmagv}
70 \text{END}

71 \text{goede_node \leftarrow \text{end point of outs[2]} \text{of goede_node}}
72 \hspace{1em} \text{JOIN goede_node TO route of tmagv} \hspace{1em} @\text{oversteken}
73 \text{goede_node \leftarrow \text{end point of outs[2]} \text{of goede_node}}
74 \hspace{1em} \text{JOIN goede_node TO route of tmagv} \hspace{1em} @\text{oversteken}
75 \hspace{1em} \text{IF tmx>\text{CEIL(tmy/10)} @de agv moet terug}
76 \hspace{1em} \text{goede_node \leftarrow nodes[200+tmy+10^{*}\text{FLOOR}(tmy/20.1)+20^{*}(\text{tmy-1})\text{MODULO}(2))]
77 \hspace{1em} \text{JOIN goede_node TO route of tmagv}
78 \text{GOTO later}
79 \text{END}

80 \text{IF CEIL(tmx/6)=3 \& CEIL(tmy/60)=2 @tweede ring pakken}
81 \hspace{1em} \text{GOTO Tweede_ring_pakken}
82 \text{END}

83 \text{IF CEIL(tmx/6)>CEIL(tmy/60)}
84 \hspace{1em} \text{IF CEIL(tmx/6)=2 \& CEIL(tmy/60)=1 @tweede ring pakken}
85 \hspace{1em} \text{GOTO Tweede_ring_pakken}
86 \hspace{1em} \text{END}

87 \hspace{1em} \text{IF CEIL(tmx/6)=3 \& CEIL(tmy/60)=2 @tweede ring pakken}
Line 152 tweede ring pakken refers to line 74

Line 130 to the end deals with the even rgv's that take the bottom route at the terminal

Line 132 to 159 deals with a rgv at a portainerlocation that has to go to a stack located behind that portainerlocation

Line 163 to 189 deals with a rgv at a portainerlocation behind berth 2 that has to go to a stack located behind berth 1 or berth 3
GOTO Tweede_ring_pakken
END

END

IF (agvnumber OF tmagv) MODULO(2)=0 @AGV is even, dus onderlangs
GOTO later
IF tmx=CEIL(tmy/10) @achter de portainer in het stack
@de portainer heeft al een entrance node gegeven
@dit is een andere dan bij de oneven agv's @
goede_node-FIRST node IN entrance OF tmagv
JOIN goede_node TO route OF tmagv
@dit is dezelfde als de entrance node

IF (tmx) MODULO(2)=1 @de agv moet heen

goede_node-nodes[ 500+ (CEIL((CEIL(tmy/10))/2))]*2]
JOIN goede_node TO route OF tmagv
END

IF (tmx) MODULO(2)=0 @de agv moet terug

goede_node ← end_point OF outs[2] OF goede_node
JOIN goede_node TO route OF tmagv @oversteken
goede_node-nodes[ 539-(CEIL((CEIL(tmy/10))/2))]*2]
JOIN goede_node TO route OF tmagv
JOIN goede_node TO route OF tmagv
JOIN goede_node TO route OF tmagv @oversteken
END

goede_node ← nodes[ 200-(tmy-1) MODULO(10) + CEIL(tmy/20)*30]
JOIN goede_node TO route OF tmagv

GOTO goede_node TO route OF tmagv

GOTO later

@IF CEIL(tmx/6)=CEIL(tmy/60) @stack zit in dezelfde berth
IF (tmx ≥7 & tmx≤12) & (CEIL(tmy/60)=1|CEIL(tmy/60)=3)

@AGV van 2 in berth op 2 gaat terug naar stack 1 of 3 via dichbij baan
@de portainer heeft al een entrance node gegeven
@dit is een andere dan bij de oneven agv's @
goede_node-FIRST node IN entrance OF tmagv
JOIN goede_node TO route OF tmagv
@dit is dezelfde als de entrance node

IF x>(tmy/10) @de agv moet terug

goede_node ← end_point OF outs[2] OF goede_node
JOIN goede_node TO route OF tmagv @oversteken
goede_node-nodes[ 539-(CEIL((CEIL(tmy/10))/2))]*2]
JOIN goede_node TO route OF tmagv
JOIN goede_node TO route OF tmagv
JOIN goede_node TO route OF tmagv @oversteken
END

IF tmx<(tmy/10) @de agv moet heen

goede_node-nodes[ 500+ (CEIL((CEIL(tmy/10))/2))]*2]
JOIN goede_node TO route OF tmagv

END

goede_node ← nodes[ 200-(tmy-1) MODULO(10) + CEIL(tmy/20)*30]
Line 191 to 229 deals with a RGV at berth 1 that has to go to a stack located behind berth 2.

Line 234 deals with a RGV that is located at berth 2 and has to go to a stack located behind berth 3.

Goto tweede ring van onderen refers to line 196.

Line 238 deals with a RGV that is located at berth 2 and has to go to a stack located behind berth 1.

Goto tweede ring van onderen refers to line 196.

Line 244 deals with a RGV that is located at berth 3 and has to go to a stack located behind berth 2.

Goto tweede ring van onderen refers to line 196.
JOIN goede_node TO route OF tmagv
GOTO Later

END

IF CEIL(tmx/6)=1 & CEIL(tmy/60)=2 @tweede ring van onderen
@de portainer heeft al een entrance node gegeven@
@dit is een andere dan bij de oneven agv's@

tweede_ring_van_onderen:
goede_node-FIRST node IN entrance OF tmagv
JOIN goede_node TO route OF tmagv
@dit is dezelfde als de entrance node

JOIN goede_node TO route OF tmagv @oversteken

JOIN goede_node TO route OF tmagv @oversteken

IF tmx>(tmy/10) @de agv moet terug
goede_node ← end_point OF outs[2] OF goede_node
JOIN goede_node TO route OF tmagv @oversteken
JOIN goede_node-nodes[579-(CEIL((CEIL(tmy/10))/2))/2]*2]
JOIN goede_node TO route OF tmagv
JOIN goede_node ← end_point OF outs[2] OF goede_node
JOIN goede_node TO route OF tmagv @oversteken

END

IF tmx<(tmy/10) @de agv moet heen
goede_node-nodes[540+ (CEIL((CEIL(tmy/10))/2))/2]*2]
JOIN goede_node TO route OF tmagv
END

goede_node ← end_point OF outs[2] OF goede_node
JOIN goede_node TO route OF tmagv @oversteken

JOIN goede_node TO route OF tmagv @oversteken

JOIN goede_node TO route OF tmagv @oversteken

goede_node ← nodes[200-(tmy-1)MODULO(10)+(CEIL(tmy/20))/30]
JOIN goede_node TO route OF tmagv

GOTO later

END

END

IF CEIL(tmx/6)=2 & CEIL(tmy/60)=3
GOTO Tweede_ring_van_onderen

END

IF CEIL(tmx/6)=2 & CEIL(tmy/60)=1
GOTO Tweede_ring_van_onderen

END

IF CEIL(tmx/6)=3 & CEIL(tmy/60)=2
GOTO Tweede_ring_van_onderen

END

later:
249 RETURN
250
1 `agv_stack_stack` is a macro that creates the route from a stack to another stack
2 This macro works the same as the macro's `agv_berth_stack` and `avg_stack_berth`
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25 The odd and the even rgv's take different routes the odd rgv's take the top route
26 If the condition in line 26 counts the rgv has to go from a stack to a stack located
27 behind the same portainerlocation (Line 25 to 36)
28
29
30
31
32
33
34
35
36
37
38
39
40 In line 40 to 69 the rgv is located in a stack behind berth 2 and has to go a stack located behind
41 berth 1 or 3
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
goede_stack ← FIRST stack IN stacklist WITH stacknumber = yy OF tmagv

GOOESECTION ← way_out_stack OF goede_stack @agvstack OF tmagv

JOIN goede_node ← end_point OF goede_section

JOIN goede_node TO entrance OF tmagv IF entrance OF tmagv IS EMPTY

tmyy ← yy OF tmagv

tmx ← x OF tmagv

tmy ← y OF tmagv

IF easyload OF tmagv = TRUE

JOIN goede_node ← nodes[ (CEIL((tmx+0.5)/2))*30+181]

JOIN goede_node TO route OF tmagv

JOIN goede_node ← nodes[ 200-(tmy-1)MODULO(10)+CEIL((tmy/20))*30]

JOIN goede_node TO route OF tmagv

GOTO later

END

IF (agvnumber OF tmagv)MODULO(2)=1 | (AGVNUMBER OF TMAGV)MODULO(2)=0

IF CEIL(tmyy/10)=CEIL(tmy/10) #de stacks liggen in hetzelfde cluster

JOIN goede_node ← nodes[ (CEIL((tmx+0.5)/2))*30+181]

JOIN goede_node TO route OF tmagv

JOIN goede_node ← nodes[ 200+y+10*(FLOOR(tmy/20.1))+20*((tmy-1)MODULO(2))]

JOIN goede_node TO route OF tmagv

GOTO later

END

IF (tmx ≥ 7 & tmx ≤ 12) & (CEIL(tmy/60)=1 | CEIL(tmy/60)=3)

@AGV van 2 in berth op 2 gaat terug naar stack 1 of 3 via dichbij baan

IF shuffleagv OF tmagv = FALSE | firstshuffle OF tmagv = TRUE

JOIN goede_node ← nodes[ (CEIL((tmx+0.5)/2))*30+181]

JOIN goede_node TO route OF tmagv

firstshuffle OF tmagv ← FALSE

END

JOIN goede_node ← nodes[ CEIL(tmyy/20)*30+200]

JOIN goede_node TO route OF tmagv

IF (tmyy/10) ≥ (tmy/10) #de agv moet terug

JOIN goede_node ← nodes[ 118-(CEIL(((CEIL(tmy/10)+0.5)/2)-1))*2]

JOIN goede_node TO route OF tmagv

END

IF (tmyy/10) < (tmy/10) #de agv moet heen

JOIN goede_node ← end_point OF outs[2] OF goede_node

JOIN goede_node TO route OF tmagv

JOIN goede_node ← nodes[ (CEIL((CEIL(tmy/10)+0.5)/2))*2+79]

JOIN goede_node TO route OF tmagv

JOIN goede_node ← end_point OF outs[2] OF goede_node
In line 74 to 121 the rgv is located in a stack behind berth 1 or 3.

Line 123 to the end deals with the even rgv's.
JOIN goede_node TO route OF tmagv
END
goede_node-nodes[200+tmy+10*(FLOOR(tmy/20.1))+20*((tmy-1)MODULO(2))]
JOIN goede_node TO route OF tmagv
GOTO later
END

IF (tmy ≥13 | tmy≤6)
Tweede_ring_pakken:
IF shuffleagv OF tmagv = FALSE | firstshuffle OF tmagv = TRUE
goede_node ← nodes[(CEIL((tmy+0.5)/2))*30+181]
JOIN goede_node TO route OF tmagv
firstshuffle OF tmagv ← FALSE
END
goede_node ← nodes[(CEIL(tmyy/20))*30+200]
JOIN goede_node TO route OF tmagv

goede_node ← end_point OF outs[2] OF goede_node
JOIN goede_node TO route OF tmagv @oversteken
goede_node ← end_point OF outs[2] OF goede_node
JOIN goede_node TO route OF tmagv @oversteken
IF (tmyy/10)>(tmy/10) @de agv moet terug
goede_node ← nodes[78-(CEIL(((CEIL(tmy/10)+0.5)/2)-1))*2]
JOIN goede_node TO route OF tmagv
END

IF (tmyy/10)<(tmy/10) @de agv moet heen
goede_node ← end_point OF outs[2] OF goede_node
JOIN goede_node TO route OF tmagv
goede_node ← nodes[(CEIL((CEIL(tmy/10)+0.5)/2))*2+39]
JOIN goede_node TO route OF tmagv
goede_node ← end_point OF outs[2] OF goede_node
JOIN goede_node TO route OF tmagv
END
goede_node ← end_point OF outs[2] OF goede_node
JOIN goede_node TO route OF tmagv @oversteken
goede_node ← end_point OF outs[2] OF goede_node
JOIN goede_node TO route OF tmagv @oversteken
goede_node ← nodes[200+tmy+10*(FLOOR(tmy/20.1))+20*((tmy-1)MODULO(2))]
JOIN goede_node TO route OF tmagv
GOTO later
END

IF (agvnumber OF tmagv)MODULO(2)=0 @AGV is even, dus onderlangs)
@@@@@@@@@@@@@@@@
@zelfde stack@
line 125 to 139 concerns a rgv going from a stack behind a portainer location to a stack behind
the same portainer location

Line 141 to 172 concerns a rgv going from a stack behind a berth to a stack behind the same
berth

Line 174 to 216 concerns a rgv going from a stack behind berth 1 or 3 to a stack behind 2
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
goede_node ← end_point OF outs[2] OF goede_node
JOIN goede_node TO route OF tmagv

IF (tmyy/10)>(tmy/10) @de agv moet terug
    goede_node ← end_point OF outs[2] OF goede_node
    JOIN goede_node TO route OF tmagv
    goedende_node-nodes[579-(CEIL(CEIL((tmy/10))/2))]*2]
    JOIN goede_node TO route OF tmagv
    goede_node ← end_point OF outs[2] OF goede_node
    JOIN goede_node TO route OF tmagv
END

IF (tmyy/10)<(tmy/10) @de agv moet heen
    goede_node-nodes[540+ (CEIL(CEIL((tmy/10))/2))]*2]
    JOIN goede_node TO route OF tmagv
END

goede_node ← end_point OF outs[2] OF goede_node
JOIN goede_node TO route OF tmagv

GO TO later
create_stack is the macro that creates the stacks this macro is called upon by the main

way_in_stack is a attribute of the stack referring to the way in

n_in is a reference to a node, the entrance of the stack

st_x is the actual x coordinate of the stack on the terminal
st_y is the actual y coordinate of the stack on the terminal
CREATE_STACK - page 1

1
2 THIS stack ← NEW stack CALLED char
3 stacknumber OF THIS stack ← i
4 IF (i) MODULO(2) = 0
5 stackcount OF THIS stack ← 0
6 END
7 wantasqueue ← NEW SET
8 way_in_stack ← NEW section
9 way_out_stack ← NEW section
10 stack_row ← NEW SET WITH HISTORY
11 n_in ← READ FROM hconfig
12 n_out ← READ FROM hconfig
13 THIS NODE ← nodes[n_in]
14 outs[2] ← way_in_stack
15 side_track[1] ← stack_crane
16 st_x ← READ FROM hconfig
17 st_y ← READ FROM hconfig
18 end_point OF way_in_stack ← THIS stack
19 s_length OF way_in_stack ← distance_to(st_x, st_y)
20 THIS node ← nodes[n_out]
21 end_point OF way_out_stack ← THIS node
22 s_length OF way_out_stack ← distance_to(st_x, st_y)
23 JOIN THIS stack TO stacklist
24 i ← i+l
25 RETURN
26
27
superShortcut is a macro designed for an exception at the terminal. On some points at the
terminal there are more shortcuts that can be taken this extra shortcut is than
called a super shortcut
1 @ The trolley passes a section being a short cut of the main route. @
2 @ Trolleys can be delayed in this section, So, SEC_ROW is needed. @
3 @----------------------------------------------------------------------@
4
5 claimed OF cur_node ← claimed OF cur_node - 1
6 cur_sec ← outs[3] OF cur_node
7
8 WORK distance/node_speed
9
10 WORK s_length OF cur_sec/low_speed
11 cur_node ← end_point OF cur_sec
12
13 REMOVE FIRST node IN superroute OF THIS agv FROM superroute OF THIS agv
14
15 WAIT WHILE claimed OF cur_node >0
16 RETURN
1 unloadyardallocation is a macro that creates the stacks for the rgv's to deliver the unloaded containers
2
3 4
5 tmtotalim is the total of containers left on the vessel that have to be unloaded
6 tmportloc is the portlocation the rgv is working for
7 im40 of a vessel is the number of 40 feet import containers
8 tmres40im is the percentage 40 feet importcontainers of all the import containers
9 tmrestotim is the percentage import containers still to be unloaded of the whole
10
11 12
13 In line 15 to 43 first the different kind of container, 40 ft or 20 ft is chosen. As a result of the kind
14 of container the stack is chosen
15 If w1 <tmres40in, the rgv gets a 40 ft container Line 15
16
17 In line 16 to 18 the lm40 container = 0 this means the
18
19 im40 of this vessel becomes im40 -1, one 40 feet container is deducted this is a counter
20
21 22
23 24
25 26
27 28 If the condition in line 30 counts the rgv will receive a 20 ft container
29
30 31
32 33
34 35
36 37
38 39
40 41
42 43 Line 44 to 52 concern the shuffling of the containers
44
45 46
47 48
49
50 51
52 53
54 55
56 57
58 59
60 61
62
begin;

3 tnx ← x OF tmagv
4
5 tntotalim ← (im40 OF tmship + im20 OF tmship)
6 tmportloc ← FIRST portloc IN portlocations WITH portlocnumber = tmx
7
8 tnmres40im ← (im40 OF tmship)/tntotalim
9 tnmrestotim ← (im40 OF tmship + im20 OF tmship)/tntotalim
10
11 w1 ← SAMPLE FROM unif
12 w2 ← SAMPLE FROM unif
13
15 IF w1 < tnmres40im
16 IF im40 OF tmship = 0 & tntotalim > 0
17 END
18 GOTO begin
19 im40 OF tmship ← im40 OF tmship - 1
20
21 y OF tmagv ← 10*(tmx-1) + →
22 (tmx)MODULO(2)*((teller40im OF tmportloc)MODULO(6)+6)+ →
23 ((tmx)-1)MODULO(2)*((teller40im OF tmportloc)MODULO(6)+5)
24
25 teller40im OF tmportloc ← teller40im OF tmportloc+ 2
26 cont20ft OF tmagv ← FALSE
27 END
28
29
30 IF w1 ≥ tnmres40im & w1 < tnmrestotim
31 IF im20 OF tmship = 0 & tntotalim > 0
32 END
33 GOTO begin
34 im20 OF tmship ← im20 OF tmship - 1
35
36 y OF tmagv ← 10*(tmx - 1) + →
37 ((tmx)-1)MODULO(2)*((teller20im OF tmportloc)MODULO(4)+1)+ →
38 (tmx)MODULO(2)*((teller20im OF tmportloc)MODULO(4)+2)
39
40 teller20im OF tmportloc ← teller20im OF tmportloc+ 2
41 cont20ft OF tmagv ← TRUE
42 END
43
44 /* A shuffle destination berthnumber is given */
45
46 w2 ← FLOOR((CEIL(w2*2)) * 0.5 + (CEIL(w2*2))) @w2 = 1 of 3
47 IF agvberthnb OF tmagv = 2
48 sbnr OF tmagv ← w2
49 END
50 IF agvberthnb OF tmagv = 1 | agvberthnb OF tmagv = 3
51 sbnr OF tmagv ← 2
52 END
53
54
55 RETURN
1 Loadyardallocation is a macro that creates the stacks where the rgv's have to get their containers

13 tmtotalex is the total of containers to be loaded unto the vessel - total already loaded

16 tex is the total of containers to be loaded unto the vessel

34 The odd rgv's take the top route of the terminal line 33 to 93

38 If this condition counts the rgv has to get a forty foot container line 38

49 In line 49 to 56 the actual stack the is chosen

61 If this condition counts the rgv has to get a twenty foot container line 61
begin:

yy OF tmagv ← y OF tmagv

tmx ← x OF tmagv
tmy ← y OF tmagv
tmyy ← yy OF tmagv

10
timportloc ← FIRST portloc IN portlocations WITH portlocnumber = tmx

11 IF tmtotalex = 0
12 RETURN
13 END
14 IF tex OF tmship = 0
15 RETURN
16 END
17

19 tmres40ex ← (ex40 OF tmship)/tmtotalex
20 tmrestotex ← (ex40 OF tmship + ex20 OF tmship)/tmtotalex

21 w1 ← SAMPLE FROM unif
22 w2 ← SAMPLE FROM unif
23 w3 ← SAMPLE FROM unif
24 w4 ← SAMPLE FROM unif
25 w5 ← SAMPLE FROM unif
26

28 IF inplacecounter[ agvberthnb OF tmagv] > 0
29 GOTO easyloading
30 END
31
32
33
34 @-- Oneven AGV's gaan bovenlangs en kunnen een 40' of 20' container krijgen --@
35

37 IF ((agvnumber OF tmagv)-1)MODULO(2) = 0 |((agvnumber OF tmagv)-1)MODULO(2)=1
38 IF w1 < tmres40ex
39 IF ex40 OF tmship = 0 & tmtotalex > 0
40 GOTO begin
41 END
42 ex40 OF tmship ← ex40 OF tmship - 1
43 w2 ← SAMPLE FROM unif
44 w2 ← CEIL(w2*2)
45 w3 ← CEIL(w3*6)+6
46 w4 ← CEIL(w4*3)
47 w5 ← CEIL(w5*4)+2
48
49 @de bovenste is om de ptloc te bepalen
50 tmy ← ((CEIL(tm/6))MODULO(2))*w3 + →
51 ((CEIL(tm/6)+1)MODULO(2))*((w5)+((w2-1)*10))
52 w2 ← SAMPLE FROM unif
53 w2 ← CEIL(w2*2)
54 @de onderste is om het stack te bepalen
55 y OF tmagv ← ((tmy)-1)*10 + →
56 ((2*w4)+(((tmy)-1)MODULO(2))+3)
57
58 cont20ft OF tmagv ← FALSE
59 END
60 IF w1 ≥ tmres40ex & w1 < tmrestotex
61 IF ex20 OF tmship = 0 & tmtotalex > 0
In line 72 to 79 the actual stack is chosen.

Line 95 to 152 is about the even rgy's which take the bottom route.

Line 100 if this condition counts the rgy has to get a 40 foot container.

In line 110 to 117 the stack is chosen to get the 40 ft container.

Line 122 if this condition counts a twenty foot container has to be retrieved from the stack.
GOTO begin
END
ex20 OF tmship = ex20 OF tmship - 1
w2 ← CEIL(w2*2)
w3 ← CEIL(w3*6)+6
w4 ← CEIL(w4*2)
w5 ← CEIL(w5*4)+2
@de bovenste is om de ptloc te bep@
tmy ← ((CEIL(tm/6))MODULO(2))*w3 + →
((CEIL(tm/6)+1)MODULO(2))*((w5)+((w2-1)*10))
w2 ← SAMPLE FROM unif
w2 ← CEIL(w2*2)
@de onderste is om het stack te bep@
y OF tmavg ← ((tmy)-1)*10 + →
(FLOOR((w4*0.5)*(((tmy)+1)MODULO(2)+3)))
cont20ft OF tmavg ← TRUE
END
IF easyload OF tmavg =TRUE
uneasy OF tmavg ← TRUE
easyload OF tmavg ← FALSE
END
easyload OF tmavg ← FALSE
GOTO provided
END
IF (agvnumber OF tmavg)MODULO(2) = 0
IF w1 < tmres40ex
IF ex40 OF tmship = 0 & tmtotalex > 0
GOTO begin
END
ex40 OF tmship = ex40 OF tmship - 1
w2 ← CEIL(w2*2)
w3 ← CEIL(w3*6)+6
w4 ← CEIL(w4*3)
w5 ← CEIL(w5*4)+2
@de bovenste is om de ptloc te bep@
tmy ← ((CEIL(tm/6))MODULO(2))*w3 + →
((CEIL(tm/6)+1)MODULO(2))*((w5)+((w2-1)*10))
w2 ← SAMPLE FROM unif
w2 ← CEIL(w2*2)
@de onderste is om het stack te bep@
y OF tmavg ← ((tmy)-1)*10 + →
((2*w4)+((tmy)MODULO(2))+3)
cont20ft OF tmavg ← FALSE
END
IF w1 ≥ tmres40ex & w1 < tmrestotex
IF ex20 OF tmship = 0 & tmtotalex > 0
GOTO begin
In line 131 to 152 the stack is chosen to get the twenty foot container.

Easy loading concerns loading where the containers are already in the stacks behind the right berth, this is only possible with shuffling.

Line 159 to 167 if this condition counts a 40 ft container has to be retrieved.

Line 168 to 176 if this condition counts a 20 ft container has to be retrieved.

Inplacecounter is the counter that counts the containers that are inplace behind that specific.
LOADYARD ALLOCATION - page 3

125 END
126 ex20 OF tmship ← ex20 OF tmship - 1
127 w2 ← CEIL(w2*2)
128 w3 ← CEIL(w3*6)+6
129 w5 ← CEIL(w5*4)+2
130 @de bovenste is om de ptloc te bep
131 tmy ← ((CEIL(tmx/6)) MODULO(2))*w3 + →
132 ((CEIL(tmx/6)+1) MODULO(2))*((w5)+((w2-1)*10))
133 w2 ← SAMPLE FROM unif
134 w2 ← CEIL(w2*2)
135 @de onderste is om het stack te bep
136 y OF tmagv ← ((tmy)-1)*10 + →
137 (FLOOR((w2*0.5)*((tmy) MODULO(2)+3)))
138 cont20ft OF tmagv ← TRUE
139 END
140 IF easyload OF tmagv =TRUE
141 uneasy OF tmagv ← TRUE
142 easyload OF tmagv ← FALSE
143 END
144 easyload OF tmagv ← FALSE
145 END
146 GOTO provided
147 EASYLOADING:
148 "------------------------------"
149 "-- Voorgeshuifelde containers worden EERST geladen van achter de portainer --86"
150 "------------------------------"
151 IF w1 < tmres40ex
152 IF ex40 OF tmship = 0 & tmtoalex > 0
153 GOTO begin
154 END
155 ex40 OF tmship ← ex40 OF tmship - 1
156 w2 ← CEIL(w2*3)
157 y OF tmagv ← ((tmx)-1)*10 +((2*w2)+((tmx) MODULO(2))+3)
158 cont20ft OF tmagv ← FALSE
159 END
160 IF w1 ≥ tmres40ex & w1 < tmres40ex
161 IF ex20 OF tmship = 0 & tmtoalex > 0
162 GOTO begin
163 END
164 ex20 OF tmship ← ex20 OF tmship - 1
165 w2 ← CEIL(w2*2)
166 y OF tmagv ← ((tmx)-1)*10 + FLOOR((w2*0.5)*((tmx) MODULO(2)+3))
167 cont20ft OF tmagv ← TRUE
168 END
169 INPLACECOUNTER[agvberthnb OF tmagv] ← INPLACECOUNTER[agvberthnb OF tmagv] - 1
170 IF INPLACECOUNTER[agvberthnb OF tmagv] = 0
171 ACTIVATE agvmaster FROM begin IN agvmastrmod
172 END
173 EASYLOAD OF tmagv ← TRUE
GOTO easyloaded

provided:

@-- calculate the number of containers left in the stange berth --@
@-- for the home berth --@

sbnr OF tmagv - CEIL((y OF tmagv)/60)
fromtoh sbnr OF tmagv,agvberthnb OF tmagv)←fromtoh sbnr OF tmagv,agvberthnb OF .

easyloaded:

RETURN

WAIT
1 **conversation** is the macro that creates the control box
2 **control box** is the box that appears when starting the run this enables the user to check the
3 stacking cranes and some nodes after the run
@ ask user for configuration data @
@-----------------------------------@

4 control ← NEW WINDOW CALLED "Control"
5 SPECIFY control ORIGIN(10,120) UNITS(5,5)
6 b_ok ← NEW WOB
7 JOIN b_ok TO control
8 n ← _wob_button(b_ok, 0, 0, 10, 5, 14, 1, 100, "OK")
9
10 @ shortcut_node @
11 prm_mn ← NEW WOB
12 str_mn ← NEW WOB
13 n ← _wob_string(str_mn, 0, 20, 10, 0, 10, " 508", 7)
14 n ← _wob_prompt(prm_mn, 10, 20, 1, "shortcut1")
15 JOIN prm_mn TO control
16 JOIN str_mn TO control
17
18 prm_maxn ← NEW WOB
19 str_maxn ← NEW WOB
20 n ← _wob_string(str_maxn, 0, 14, 10, 0, 10, " 260", 7)
21 n ← _wob_prompt(prm_maxn, 10, 14, 1, "shortcut2")
22 JOIN prm_maxn TO control
23 JOIN str_maxn TO control
24
25 prm_Mt ← NEW WOB
26 str_Mt ← NEW WOB
27 n ← _wob_string(str_Mt, 25, 20, 10, 0, 10, " 453", 7)
28 n ← _wob_prompt(prm_Mt, 35, 20, 1, "shortcut3")
29 JOIN prm_Mt TO control
30 JOIN str_Mt TO control
31
32 prm_m ← NEW WOB
33 str_m ← NEW WOB
34 n ← _wob_string(str_m, 25, 14, 10, 0, 10, " 512", 7)
35 n ← _wob_prompt(prm_m, 35, 14, 1, "shortcut4")
36 JOIN prm_m TO control
37 JOIN str_m TO control
38
39
40
41 @ cranes @
42
43 prm_01 ← NEW WOB
44 str_01 ← NEW WOB
45 n ← _wob_string(str_01, 0, 8, 10, 3, 13, " 21", 7)
46 n ← _wob_prompt(prm_01, 10, 8, 1, "stack1")
47 JOIN prm_01 TO control
48 JOIN str_01 TO control
49
50 prm_02 ← NEW WOB
51 str_02 ← NEW WOB
52 n ← _wob_string(str_02, 25, 8, 10, 0, 13, " 22", 7)
53 n ← _wob_prompt(prm_02, 35, 8, 1, "stack2")
54 JOIN prm_02 TO control
55 JOIN str_02 TO control
56
57 prm_03 ← NEW WOB
58 str_03 ← NEW WOB
59 n ← _wob_string(str_03, 50, 8, 10, 3, 13, " 23", 7)
60 n ← _wob_prompt(prm_03, 60, 8, 1, "stack3")
61 JOIN prm_03 TO control
62 JOIN str_03 TO control
63   prm_04 - NEW WOB
64   str_04 - NEW WOB
65   n - _wob_string(str_04,75,8,10,0,13," 24",7)
66   n - _wob_prompt(prm_04,85,8,1,"stack4")
67   JOIN prm_04 TO control
68   JOIN str_04 TO control
69
70   prm_05 - NEW WOB
71   str_05 - NEW WOB
72   n - _wob_string(str_05,0,-8,10,3,13," 25",7)
73   n - _wob_prompt(prm_05,10,-8,1,"stack5")
74   JOIN prm_05 TO control
75   JOIN str_05 TO control
76
77   prm_06 - NEW WOB
78   str_06 - NEW WOB
79   n - _wob_string(str_06,25,-8,10,0,13," 26",7)
80   n - _wob_prompt(prm_06,35,-8,1,"stack6")
81   JOIN prm_06 TO control
82   JOIN str_06 TO control
83
84   prm_07 - NEW WOB
85   str_07 - NEW WOB
86   n - _wob_string(str_07,50,-8,10,3,13," 27",7)
87   n - _wob_prompt(prm_07,60,-8,1,"stack7")
88   JOIN prm_07 TO control
89   JOIN str_07 TO control
90
91   prm_08 - NEW WOB
92   str_08 - NEW WOB
93   n - _wob_string(str_08,75,-8,10,0,13," 28",7)
94   n - _wob_prompt(prm_08,85,-8,1,"stack8")
95   JOIN prm_08 TO control
96   JOIN str_08 TO control
97
98   prm_09 - NEW WOB
99   str_09 - NEW WOB
100  n - _wob_string(str_09,0,-16,10,3,13," 29",7)
101  n - _wob_prompt(prm_09,10,-16,1,"stack9")
102  JOIN prm_09 TO control
103  JOIN str_09 TO control
104
105   prm_10 - NEW WOB
106   str_10 - NEW WOB
107   n - _wob_string(str_10,25,-16,10,0,13," 30",7)
108   n - _wob_prompt(prm_10,35,-16,1,"stack10")
109   JOIN prm_10 TO control
110   JOIN str_10 TO control
111
112   SHOW control AT(20,20) SIZE(500,340)
113
114
115
116   rep_cntr:
117   n - _window_response(control)
118   n - _wob_string_value(str_mn,<char9>)
119   REPEAT FROM rep_cntr IF numeric_control(<char9>) = FALSE
120   mn - <char9>
121
122   n - _wob_string_value(str_maxn,<char9>)
123   REPEAT FROM rep_cntr IF numeric_control(<char9>) = FALSE
124
mnb ← char9

n ← _wob_string_value(str_01, <char9>)
REPEAT FROM rep_cntr IF numeric_control(<char9>) = FALSE
   c1 ← char9

n ← _wob_string_value(str_02, <char9>)
REPEAT FROM rep_cntr IF numeric_control(<char9>) = FALSE
   c2 ← char9

n ← _wob_string_value(str_03, <char9>)
REPEAT FROM rep_cntr IF numeric_control(<char9>) = FALSE
   c3 ← char9

n ← _wob_string_value(str_04, <char9>)
REPEAT FROM rep_cntr IF numeric_control(<char9>) = FALSE
   c4 ← char9

n ← _wob_string_value(str_05, <char9>)
REPEAT FROM rep_cntr IF numeric_control(<char9>) = FALSE
   c5 ← char9

n ← _wob_string_value(str_06, <char9>)
REPEAT FROM rep_cntr IF numeric_control(<char9>) = FALSE
   c6 ← char9

n ← _wob_string_value(str_07, <char9>)
REPEAT FROM rep_cntr IF numeric_control(<char9>) = FALSE
   c7 ← char9

n ← _wob_string_value(str_08, <char9>)
REPEAT FROM rep_cntr IF numeric_control(<char9>) = FALSE
   c8 ← char9

n ← _wob_string_value(str_09, <char9>)
REPEAT FROM rep_cntr IF numeric_control(<char9>) = FALSE
   c9 ← char9

n ← _wob_string_value(str_10, <char9>)
REPEAT FROM rep_cntr IF numeric_control(<char9>) = FALSE
   c10 ← char9

n ← _wob_string_value(str_m, <char9>)
REPEAT FROM rep_cntr IF numeric_control(<char9>) = FALSE
   nm ← char9

n ← _wob_string_value(str_mt, <char9>)
REPEAT FROM rep_cntr IF numeric_control(<char9>) = FALSE
   bmn ← char9

RETURN
Preclaimnodes is a macro called upon by the agv responsible for claiming nodes when more
than one node has to be claimed in order to enter the main track

line 1 to 3 a rgv wants to enter a main track but it has to wait while the entrance node is claimed

Line 8 to 37 as many nodes will be claimed as needed for the rgv to enter the main track

In line 10 to 15 there are some exceptions, in the case of these nodes (line 10) only one node has
to be claimed because they lie at the end of the main track (node 231,261,291 etc)

line 22 to 28 deals with an exception at the end of a main track

claimset is a set of nodes that have to be claimed if the length of the sections between the nodes
is 30 it is possible for the rgv to enter the main track else another node has to be claimed in order
to extend the claimedlength

claimedlength is the length the rgv needs to enter the main track

In line 42 to 46 the set is emptied and the claimed nodes are unclaimed

The rgv returns to the agymodule where it came from line 48 and enters the main track

The rgv has to wait while claimed of claim_node is not 0
go to again in line 8 from there another node will be claimed until the claimedlength is >=30 meter

1 IF claimed OF entrance_node ≠ 0
2    WAIT WHILE claimed OF entrance_node ≠ 0
3 END
4
5 claimedlength OF THIS agv ← 0
6 b ← node_number OF entrance_node
7
8 again:
9
10 IF b=231|b=261|b=291|b=321|b=351|b=381|b=411|b=441|b=471
11    claim_node ← FIRST node IN node_list WITH node_number = b
12    nodeagvnb OF claim_node ← agvnumber OF THIS agv
13    claimed OF claim_node ← claimed OF claim_node + 1
14    ACTIVATE claim_node FROM begin IN nodemod
15    GOTO empty_the_set
16 END
17
18 b←b-1
19 claim_node ← FIRST node IN node_list WITH node_number = b
20 claim_sec ← outs[1] OF claim_node
21
22 IF claim_sec IS NONE
23    claim_node ← FIRST node IN node_list WITH node_number = b+1
24    nodeagvnb OF claim_node ← agvnumber OF THIS agv
25    claimed OF claim_node ← claimed OF claim_node + 1
26    ACTIVATE claim_node FROM begin IN nodemod
27    GOTO empty_the_set
28 END
29
30 IF claimed OF claim_node = 0
31    claimed OF claim_node ← claimed OF claim_node + 1
32    claimedlength OF THIS agv ← s_length OF claim_sec + claimedlength OF THIS agv
33    JOIN claim_node TO claimset
34
35 IF claimedlength < 30
36    GOTO again
37 END
38
39 empty_the_set:
40 Le-LENGTH OF claimset
41 FOR i ← 1 TO Le
42    claim_node ← FIRST node IN claimset
43    REMOVE FIRST node IN claimset FROM claimset
44    claimed OF claim_node ← claimed OF claim_node - 1
45 END
46
47 RETURN
48 END
49
50 WAIT WHILE claimed OF claim_node ≠ 0
51 GOTO again
agv_uneasy_b_e is a macro that creates a route from the berth to a stack
This macro is called upon by the trafficmaster in line 282 in the trafficmastermodule, this case is
the transition between loading a vessel and shuffling and only counts for a even rgv

Line 16 and 17 the exit of the portainerlocation becomes a point for the route of the rgv this means
the rgv has to leave the main track the moment it enters the main track behind the portainer
If this condition counts the rgv is at berth 2 and has to go to a stack behind berth 1 or the rgv is
at berth 3 in both cases the rgv has to go back line 19 to 50

If this condition counts the rgv os at berth 1 and has to go to a stack located behind berth 2 or the
rgv is at berth 2 and has to go to a stack located behind berth 3 line 52 to 88
begin:
3 tmy-y OF tmagv
4 tmx-x OF tmagv
5 tmyy-yy OF tmagv
6
7 agvstack-FIRST stack IN stacklist WITH stacknumber = tmy
8
9
10 IF shuffleagv OF tmagv = TRUE
11     GOTO uneasy_first_shuffle
12 END
13
14 unease OF tmagv ← FALSE
15
16 cur_node ← nodes[ CEIL(tmx/2)*30+200]
17 JOIN cur_node TO route OF tmagv
18
19 IF CEIL(tmx/6)=3\(\) \& CEIL(tmy/60)=1 @de agv gaat terug
20
21     cur_node ← nodes[ 118-(CEIL(((tmx-1)+0.5)/2)-1))*2]
22     JOIN cur_node TO route OF tmagv
23
24     cur_node-nodes[ 213+CEIL((tmx-2)/2)*30]
25     JOIN cur_node TO route OF tmagv
26
27     cur_node ← end_point OF outs[2] OF cur_node
28     JOIN cur_node TO route OF tmagv  @oversteken
29
30     cur_node ← end_point OF outs[2] OF cur_node
31     JOIN cur_node TO route OF tmagv  @oversteken
32
33     cur_node ← end_point OF outs[2] OF cur_node
34     JOIN cur_node TO route OF tmagv  @oversteken
35
36     cur_node-nodes[ 579-(CEIL((CEIL(tmy/10))/2))*2]
37     JOIN cur_node TO route OF tmagv
38
39     cur_node ← end_point OF outs[2] OF cur_node
40     JOIN cur_node TO route OF tmagv
41
42     cur_node ← end_point OF outs[2] OF cur_node
43     JOIN cur_node TO route OF tmagv  @oversteken
44
45     cur_node ← end_point OF outs[2] OF cur_node
46     JOIN cur_node TO route OF tmagv  @oversteken
47
48     cur_node ← nodes[ 200-(tmy-1)MODULO(10)+(CEIL(tmy/20))*30]
49     JOIN cur_node TO route OF tmagv
50 END
51
52 IF CEIL(tmx/6)=1 \& CEIL(tmx/6)=2 \& CEIL(tmy/60)=3 @de agv gaat heen
53
54     cur_node ← end_point OF outs[2] OF cur_node
55     JOIN cur_node TO route OF tmagv
56
57     cur_node ← end_point OF outs[2] OF cur_node
58
59     cur_node ← end_point OF outs[1] OF cur_node
60     JOIN cur_node TO route OF tmagv
61
62     cur_node ← end_point OF outs[2] OF cur_node
uneasy first shuffle creates a route for the rgv to the first shuffle location line 91 to 123
JOIN cur_node TO route OF tmagv

cur_node = nodes[213+CEIL((tmx)/2)*30]
JOIN cur_node TO route OF tmagv

cur_node = end_point OF outs[2] OF cur_node
JOIN cur_node TO route OF tmagv

cur_node = end_point OF outs[2] OF cur_node
JOIN cur_node TO route OF tmagv

cur_node = nodes[540+(CEIL((CEIL(tmy/10))/2))*2]
JOIN cur_node TO route OF tmagv

cur_node = end_point OF outs[2] OF cur_node
JOIN cur_node TO route OF tmagv  \@oversteken

cur_node = end_point OF outs[2] OF cur_node
JOIN cur_node TO route OF tmagv  \@oversteken

cur_node = nodes[200-(tmy-1)MODULO(10)+(CEIL(tmy/20))*30]
JOIN cur_node TO route OF tmagv

END

RETURN

uneasy_first_shuffle:

IF (tmx)MODULO(2) = 0

JOIN cur_node TO route OF tmagv

cur_node = end_point OF outs[2] OF cur_node
JOIN cur_node TO route OF tmagv

cur_node = end_point OF outs[2] OF cur_node
JOIN cur_node TO route OF tmagv

cur_node = end_point OF outs[2] OF cur_node
JOIN cur_node TO route OF tmagv

cur_node = nodes[200-(tmy-1)MODULO(10)+(CEIL(tmy/20))*30]
JOIN cur_node TO route OF tmagv

END

IF (tmx)MODULO(2) = 1

JOIN cur_node TO route OF tmagv

cur_node = nodes[500+(tmx)+1]
JOIN cur_node TO route OF tmagv

cur_node = nodes[200-(tmy-1)MODULO(10)+(CEIL(tmy/20))*30]
JOIN cur_node TO route OF tmagv

END

124
125 RETURN
1 yardshuffleallocation is the macro that creates the locations for the rgv to retrieve containers
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16 In line 16 to 69
17
18 Line 20 the second condition says the rgv is at his portainerlocation or in the stack behind its
19 own portainerlocation
20 firstshuffle is an attribute of the rgv and indicates this rgv is shuffling for the first time
21
22
23
24
25 Depending of the kind of container a stack is chosen where to get the first shuffle
26 container line 26 to 36 deals with the 20 ft containers
27
28
29
30
31
32
33
34
35
36
37 Line 37 to 47 deals with the 40 ft containers
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52 sbnr is the stack berth number and identifies behind which berth the stacks are located where the
53 rgv has to go. If the rgv is at berth 1 or 3 (condition in line 52) the rgv has to go berth 2
54
55 If the rgv is at berth 2 it has to go to berth 1 or 3 line 55
1 begin:
2
3 shuffleagv OF tmagv ← TRUE
4
5 tmy ← y OF tmagv
6 tmx ← x OF tmagv
7 tmxy ← yy OF tmagv
8
9 w1 ← SAMPLE FROM unif
10 w2 ← SAMPLE FROM unif
11 w3 ← SAMPLE FROM unif
12 w4 ← SAMPLE FROM unif
13 w5 ← SAMPLE FROM unif
14 w6 ← SAMPLE FROM unif
15
tmxy ← (CEIL(tmxy/10))) & firstshuffle OF tmagv = TRUE
16
17 IF loadedagv OF tmagv = FALSE & (yard OF tmagv = FALSE |→
18 tmyx = CEIL(tmxy/10)) & firstshuffle OF tmagv = TRUE
19
20 yy OF tmagv ← y OF tmagv
21
22 IF cont20shuf OF tmagv = TRUE
23 IF (tmx) MODULO(2)=0
24 w1 ← CEIL(w1*4)
25 y OF tmagv ← (tmx-1)*10 + ((w1) MODULO(2))*(w1) + ((w1+1) MODULO(2))*(w1)
26 END
27 IF (tmx) MODULO(2)=1
28 w1 ← CEIL(w1*4)
29 y OF tmagv ← (tmx-1)*10 + ((w1+1) MODULO(2))*(w1) + ((w1) MODULO(2))*(w1)
30 END
31 cont20ft OF tmagv ← TRUE
32
33 IF cont20shuf OF tmagv = FALSE
34 IF (tmx) MODULO(2)=0
35 w1 ← CEIL(w1*6)+4
36 y OF tmagv ← (tmx-1)*10 + ((w1) MODULO(2))*(w1) + ((w1+1) MODULO(2))*(w1)
37 END
38 IF (tmx) MODULO(2)=1
39 w1 ← CEIL(w1*6)+4
40 y OF tmagv ← (tmx-1)*10 + ((w1+1) MODULO(2))*(w1) + ((w1) MODULO(2))*(w1)
41 END
42 cont20ft OF tmagv ← FALSE
43
44 tmyx ← tmy
45 tmy ← y OF tmagv
46
47 IF CEIL((y OF tmagv)/60) MODULO(2)=1
48 sbnr OF tmagv ← 2
49 END
50 IF CEIL((y OF tmagv)/60) MODULO(2)=0
51 w2 ← CEIL(w2*2)
52 sbnr OF tmagv ← FLOOR(w2*1.5) @1 or 3
53 END
54
55 GOTO ready
56 END
69 **line 69 to 90** is about collecting a container under the crane at its own stack not for the first time
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92 **Line 92 to 189** is about the delivering of a container from one stack to another starting at a stack
93 behind the rgv's own portainerlocation
94
95
96 If the conditions in **line 96** count the rgv is loaded in a stack behind its own portainerlocation
97
98 if sbnr=0 it indicates the rgv is back in is own stack ready to go to another stack **Line 98 to 138**
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
IF loadedagv OF tmagv = FALSE & tmx = (CEIL(tmy/10)) & firstshuffle OF tmagv = FALSE & sbnr OF tmagv = 0

yy OF tmagv ← y OF tmagv
w1 ← CEIL(w1*9)
w2 ← CEIL(w2*2)

IF (tmy) MODULO(10) ≤ 4 & (tmy) MODULO(10) ≠ 0
    cont20ft OF tmagv ← TRUE
END
IF (tmy) MODULO(10) ≥ 5 | (tmy) MODULO(10) = 0
    cont20ft OF tmagv ← FALSE
END

IF CEIL((y OF tmagv)/60) MODULO(2) = 1
    sbnr OF tmagv ← 2
END
IF CEIL((y OF tmagv)/60) MODULO(2) = 0
    sbnr OF tmagv ← FLOOR(w2*1.5) @1 or 3
END

GOTO ready
END

IF sbnr OF tmagv = 0
    yy OF tmagv ← y OF tmagv
w1 ← CEIL(w1*2)
w2 ← CEIL(w2*2)
w3 ← CEIL(w3*6)+6
w4 ← CEIL(w4*3)
w5 ← CEIL(w5*4)+2

trm ← ((CEIL(tmx/6)) MODULO(2)) * w3 + 
     ((CEIL(tmx/6)+1) MODULO(2)) * ((w5 + ((w2-1)*10))

sbnr OF tmagv ← CEIL((tmy)/6)

IF cont20ft OF tmagv = TRUE
    IF (agvnnumber OF tmagv) MODULO(2) = 1 @ bovenlangs
        y OF tmagv ← ((tmy)-1)*10 + 
                     (FLOOR((w2*0.5)* (((tmy)+1) MODULO(2)+3)))
    END
    IF (agvnnumber OF tmagv) MODULO(2) = 0 @ onderlangs
        y OF tmagv ← ((tmy)-1)*10 + 
                     (FLOOR((w2*0.5)* (((tmy) MODULO(2)+3)))
    END
GOTO almostready
END
Line 140 to 182 is almost the same as line 98 to 138 except for the fact that it is the first time for the rgy
YARDSHUFFLELOCATION – page 3

125 IF cont20ft OF tmagv = FALSE
126 IF (agvnumber OF tmagv) MODULO(2) = 1 @bovenlangs
127 y OF tmagv ← ((tmy)-1)*10 + →
128 (2*w4)+(((tmy)-1) MODULO(2))+3
129 END
130
131 IF (agvnumber OF tmagv) MODULO(2) = 0 @onderlangs
132 y OF tmagv ← ((tmy)-1)*10 + →
133 (2*w4)+(((tmy) MODULO(2))+3)
134 END
135 END
136
137 GOTO almostready
138 END
139
140 IF sbnr OF tmagv ≠ 0
141 yy OF tmagv ← y OF tmagv
142 w1 ← CEIL(w1*2)
143 w2 ← CEIL(w2*2)
144 w3 ← CEIL(w3*4)+6
145 w4 ← CEIL(w4*3)
146 w5 ← CEIL(w5*4)+2
147 w6 ← CEIL(w6*6)+6
148 IF sbnr OF tmagv = 2
149 tmy ← ((CEIL(tm/6)) MODULO(2))*w6 + →
150 ((CEIL(tm/6)+1) MODULO(2))*((w5)+((w2-1)*10))
151 END
152
153 IF sbnr OF tmagv = 1 | sbnr OF tmagv = 3
154 tmy ← w3+6-(sbnr OF tmagv) MODULO(3)*10
155 END
156
157 IF cont20ft OF tmagv = TRUE
158 IF (agvnumber OF tmagv) MODULO(2) = 1 @bovenlangs
159 y OF tmagv ← ((tmy)-1)*10 + →
160 (FLOOR((w2*0.5)*(((tmy)+1) MODULO(2)+3)))
161 END
162
163 IF (agvnumber OF tmagv) MODULO(2) = 0 @onderlangs
164 y OF tmagv ← ((tmy)-1)*10 + →
165 (FLOOR((w2*0.5)*(((tmy) MODULO(2)+3)))
166 END
167 GOTO almostready
168 END
169
170 IF cont20ft OF tmagv = FALSE
171 IF (agvnumber OF tmagv) MODULO(2) = 1 @bovenlangs
172 y OF tmagv ← ((tmy)-1)*10 + →
173 (2*w4)+(((tmy)-1) MODULO(2))+3
174 END
175
176 IF (agvnumber OF tmagv) MODULO(2) = 0 @onderlangs
177 y OF tmagv ← ((tmy)-1)*10 + →
178 (2*w4)+(((tmy) MODULO(2))+3)
179 END
180
181 GOTO almostready
182 END
183
184 almostready:
In line 190 to 230 the rgv is located in a stack, not behind its own berth and has to go back to a stack behind its own berth.
187  GOTO ready
188  END
189
190  @-- returning to own stack with or without a container  --@
191
192
193  IF yard OF tmagv = TRUE & tmx ≠ (CEIL(tmy/10))
194      yy OF tmagv ← y OF tmagv
195
196      w1 ← CEIL(w1*2)
197      w2 ← CEIL(w2*2)
198      w3 ← CEIL(w3*6)+6
199      w4 ← CEIL(w4*3)
200
201      tmx ← tmx
202      sbnr OF tmagv ← 0  @resetting the sbnr of tmagv
203
204  END
205
206  IF cont20ft OF tmagv = TRUE
207      IF (agvnumber OF tmagv)MODULO(2)=1 @oneven dus bovenlangs
208          y OF tmagv ← ((tmy)-1)*10 + →
209                (FLOOR((w2*0.5)*((tmy)+1)MODULO(2)+3))
210     END
211
212  IF (agvnumber OF tmagv)MODULO(2)=0 @even dus onderlangs
213      y OF tmagv ← ((tmy)-1)*10 + →
214                (FLOOR((w2*0.5)*((tmy)MODULO(2)+3))
215     END
216
217  END
218
219  IF cont20ft OF tmagv = FALSE
220     IF (agvnumber OF tmagv)MODULO(2)=1 @oneven dus bovenlangs
221          y OF tmagv ← ((tmy)-1)*10 + →
222                ((2*w4)+(((tmy)-1)MODULO(2))+3)
223     END
224
225  IF (agvnumber OF tmagv)MODULO(2)=0 @even dus onderlangs
226      y OF tmagv ← ((tmy)-1)*10 + →
227                ((2*w4)+((tmy)MODULO(2))+3)
228     END
229
230  END
231
232  RETURN
The creation chooses the cranes that have to serve the stacks. It is possible via the creation to vary the number of stacking cranes.
begin:
  yyy OF tmagv = y OF tmagv
  xxx OF tmagv = FLOOR((yyy OF tmagv-1)/20)
  zzz OF tmagv = FLOOR((yyy OF tmagv-1)/20)
  yyy OF tmagv = (y OF tmagv) - (zzz OF tmagv)*20
  IF (yyy OF tmagv) MODULO (2) = 0
    yyy OF tmagv = yyy OF tmagv - 1
  END
  @IF (yyy OF tmagv) > 10
    @yyy OF tmagv = yyy OF tmagv - 10
  @END
  yyy OF tmagv = yyy OF tmagv + 20*(xxx OF tmagv)
  avcrane OF tmagv = FIRST crane IN cranelist-
  WITH cranenumber = yyy OF tmagv
RETURN
1 Create outportations is a macro that creates the outportations
CREATE_OUTPORTATION - page 1

1 i-1
2 WHILE i<300
3   outportation[i] = NEW DATASTREAM
4   i = i + 1
5 END

6 ATTACH "outportation1.txt" TO outportation[1]
7 ATTACH "outportation2.txt" TO outportation[2]
8 ATTACH "outportation3.txt" TO outportation[3]
9 ATTACH "outportation4.txt" TO outportation[4]
10 ATTACH "outportation5.txt" TO outportation[5]
11 ATTACH "outportation6.txt" TO outportation[6]
12 ATTACH "outportation7.txt" TO outportation[7]
13 ATTACH "outportation8.txt" TO outportation[8]
14 ATTACH "outportation9.txt" TO outportation[9]
15 ATTACH "outportation10.txt" TO outportation[10]
16 ATTACH "outportation11.txt" TO outportation[11]
17 ATTACH "outportation12.txt" TO outportation[12]
18 ATTACH "outportation13.txt" TO outportation[13]
19 ATTACH "outportation14.txt" TO outportation[14]
20 ATTACH "outportation15.txt" TO outportation[15]
21 ATTACH "outportation16.txt" TO outportation[16]
22 ATTACH "outportation17.txt" TO outportation[17]
23 ATTACH "outportation18.txt" TO outportation[18]
24 ATTACH "outportation19.txt" TO outportation[19]
25 ATTACH "outportation20.txt" TO outportation[20]
26 ATTACH "outportation21.txt" TO outportation[21]
27 ATTACH "outportation22.txt" TO outportation[22]
28 ATTACH "outportation23.txt" TO outportation[23]
29 ATTACH "outportation24.txt" TO outportation[24]
30 ATTACH "outportation25.txt" TO outportation[25]
31 ATTACH "outportation26.txt" TO outportation[26]
32 ATTACH "outportation27.txt" TO outportation[27]
33 ATTACH "outportation28.txt" TO outportation[28]
34 ATTACH "outportation29.txt" TO outportation[29]
35 ATTACH "outportation30.txt" TO outportation[30]
36 ATTACH "outportation31.txt" TO outportation[31]
37 ATTACH "outportation32.txt" TO outportation[32]
38 ATTACH "outportation33.txt" TO outportation[33]
39 ATTACH "outportation34.txt" TO outportation[34]
40 ATTACH "outportation35.txt" TO outportation[35]
41 ATTACH "outportation36.txt" TO outportation[36]
42 ATTACH "outportation37.txt" TO outportation[37]
43 ATTACH "outportation38.txt" TO outportation[38]
44 ATTACH "outportation39.txt" TO outportation[39]
45 ATTACH "outportation40.txt" TO outportation[40]
46 ATTACH "outportation41.txt" TO outportation[41]
47 ATTACH "outportation42.txt" TO outportation[42]
48 ATTACH "outportation43.txt" TO outportation[43]
49 ATTACH "outportation44.txt" TO outportation[44]
50 ATTACH "outportation45.txt" TO outportation[45]
51 ATTACH "outportation46.txt" TO outportation[46]
52 ATTACH "outportation47.txt" TO outportation[47]
53 ATTACH "outportation48.txt" TO outportation[48]
54 ATTACH "outportation49.txt" TO outportation[49]
55 ATTACH "outportation50.txt" TO outportation[50]
56 ATTACH "outportation51.txt" TO outportation[51]
57 ATTACH "outportation52.txt" TO outportation[52]
58 ATTACH "outportation53.txt" TO outportation[53]
59 ATTACH "outportation54.txt" TO outportation[54]
60 ATTACH "outportation55.txt" TO outportation[55]
61 ATTACH "outportation56.txt" TO outportation[56]
ATTACH "outportation242.txt" TO outportation [242]  
ATTACH "outportation243.txt" TO outportation [243]  
ATTACH "outportation244.txt" TO outportation [244]  
ATTACH "outportation245.txt" TO outportation [245]  
ATTACH "outportation246.txt" TO outportation [246]  
ATTACH "outportation247.txt" TO outportation [247]  
ATTACH "outportation248.txt" TO outportation [248]  
ATTACH "outportation249.txt" TO outportation [249]  
ATTACH "outportation250.txt" TO outportation [250]  
ATTACH "outportation251.txt" TO outportation [251]  
ATTACH "outportation252.txt" TO outportation [252]  
ATTACH "outportation253.txt" TO outportation [253]  
ATTACH "outportation254.txt" TO outportation [254]  
ATTACH "outportation255.txt" TO outportation [255]  
ATTACH "outportation256.txt" TO outportation [256]  
ATTACH "outportation257.txt" TO outportation [257]  
ATTACH "outportation258.txt" TO outportation [258]  
ATTACH "outportation259.txt" TO outportation [259]  
ATTACH "outportation260.txt" TO outportation [260]  
ATTACH "outportation261.txt" TO outportation [261]  
ATTACH "outportation262.txt" TO outportation [262]  
ATTACH "outportation263.txt" TO outportation [263]  
ATTACH "outportation264.txt" TO outportation [264]  
ATTACH "outportation265.txt" TO outportation [265]  
ATTACH "outportation266.txt" TO outportation [266]  
ATTACH "outportation267.txt" TO outportation [267]  
ATTACH "outportation268.txt" TO outportation [268]  
ATTACH "outportation269.txt" TO outportation [269]  
ATTACH "outportation270.txt" TO outportation [270]  
ATTACH "outportation271.txt" TO outportation [271]  
ATTACH "outportation272.txt" TO outportation [272]  
ATTACH "outportation273.txt" TO outportation [273]  
ATTACH "outportation274.txt" TO outportation [274]  
ATTACH "outportation275.txt" TO outportation [275]  
ATTACH "outportation276.txt" TO outportation [276]  
ATTACH "outportation277.txt" TO outportation [277]  
ATTACH "outportation278.txt" TO outportation [278]  
ATTACH "outportation279.txt" TO outportation [279]  
ATTACH "outportation280.txt" TO outportation [280]  
ATTACH "outportation281.txt" TO outportation [281]  
ATTACH "outportation282.txt" TO outportation [282]  
ATTACH "outportation283.txt" TO outportation [283]  
ATTACH "outportation284.txt" TO outportation [284]  
ATTACH "outportation285.txt" TO outportation [285]  
ATTACH "outportation286.txt" TO outportation [286]  
ATTACH "outportation287.txt" TO outportation [287]  
ATTACH "outportation288.txt" TO outportation [288]  
ATTACH "outportation289.txt" TO outportation [289]  
ATTACH "outportation290.txt" TO outportation [290]  
ATTACH "outportation291.txt" TO outportation [291]  
ATTACH "outportation292.txt" TO outportation [292]  
ATTACH "outportation293.txt" TO outportation [293]  
ATTACH "outportation294.txt" TO outportation [294]  
ATTACH "outportation295.txt" TO outportation [295]  
ATTACH "outportation296.txt" TO outportation [296]  
ATTACH "outportation297.txt" TO outportation [297]  
ATTACH "outportation298.txt" TO outportation [298]  
ATTACH "outportation299.txt" TO outportation [299]  
ATTACH "outportation300.txt" TO outportation [300]  
RETURN
crane cycletime is a macro that creates the cycletimes for the stacking cranes

F is the time in seconds the crane has to work
@this is where the cycle times of the stacking cranes are created

begin:

w1-SAMPLE FROM unif
w2-SAMPLE FROM UNIF

IF w1≥0.5
   w1=80+36*w2
END

IF w1<0.5
   w1=44+36*w2
END

F OF THIS CRANE = w1

RETURN
1 crane outportations is a macro that creates the outportations for the stacking cranes every crane
2 gets its own outportation if it has been doing something
G-CRANENUMBER OF THIS CRANE

CONVERT (cranenumbe OF THIS crane) TO outstring FIELDDLENGTH 12
WRITE outstring TO OUTPORTATION[ G] WITH IMAGE a(12)
CONVERT NOW TO OUTSTRING FIELDDLENGTH 12 DECIMALS 3
WRITE outstring TO outportation[ G] WITH IMAGE A(12)
CONVERT f OF THIS crane TO OUTSTRING FIELDDLENGTH 12 DECIMALS 3
WRITE outstring TO outportation[ G] WITH IMAGE A(12)

RETURN
creating sections is a macro that creates sections and attach these sections to windws in this way all the queues at the sections can be monitored
CREATING SECTIONS - page 1

1 newsection-FIRST section IN sections WITH sec_row_number = mn
2 secrow[ 1] - NEW WINDOW CALLED "noderow1"
3 ATTACH HISTORY OF sec_row OF newsection TO secrow[ 1]
4
5 newsection-FIRST section IN sections WITH sec_row_number = mnb
6 secrow[ 2] - NEW WINDOW CALLED "noderow2"
7 ATTACH HISTORY OF sec_row OF newsection TO secrow[ 2]
8
9 newsection-FIRST section IN sections WITH sec_row_number = bmn
10 secrow[ 3] - NEW WINDOW CALLED "noderow3"
11 ATTACH HISTORY OF sec_row OF newsection TO secrow[ 3]
12
13 newsection-FIRST section IN sections WITH sec_row_number = nm
14 secrow[ 4] - NEW WINDOW CALLED "noderow4"
15 ATTACH HISTORY OF sec_row OF newsection TO secrow[ 4]
16
17 newsection-FIRST section IN sections WITH sec_row_number = 116
18 secrow[ 145] - NEW WINDOW CALLED "noderow116"
19 ATTACH HISTORY OF sec_row OF newsection TO secrow[ 145]
20
21 newsection-FIRST section IN sections WITH sec_row_number = 114
22 secrow[ 146] - NEW WINDOW CALLED "noderow114"
23 ATTACH HISTORY OF sec_row OF newsection TO secrow[ 146]
24
25 newsection-FIRST section IN sections WITH sec_row_number = 112
26 secrow[ 147] - NEW WINDOW CALLED "noderow112"
27 ATTACH HISTORY OF sec_row OF newsection TO secrow[ 147]
28
29 newsection-FIRST section IN sections WITH sec_row_number = 110
30 secrow[ 148] - NEW WINDOW CALLED "noderow110"
31 ATTACH HISTORY OF sec_row OF newsection TO secrow[ 148]
32
33 newsection-FIRST section IN sections WITH sec_row_number = 108
34 secrow[ 149] - NEW WINDOW CALLED "noderow108"
35 ATTACH HISTORY OF sec_row OF newsection TO secrow[ 149]
36
37 newsection-FIRST section IN sections WITH sec_row_number = 106
38 secrow[ 150] - NEW WINDOW CALLED "noderow106"
39 ATTACH HISTORY OF sec_row OF newsection TO secrow[ 150]
40
41 newsection-FIRST section IN sections WITH sec_row_number = 104
42 secrow[ 151] - NEW WINDOW CALLED "noderow104"
43 ATTACH HISTORY OF sec_row OF newsection TO secrow[ 151]
44
45 newsection-FIRST section IN sections WITH sec_row_number = 102
46 secrow[ 152] - NEW WINDOW CALLED "noderow102"
47 ATTACH HISTORY OF sec_row OF newsection TO secrow[ 152]
48
49 newsection-FIRST section IN sections WITH sec_row_number = 260
50 secrow[ 153] - NEW WINDOW CALLED "noderow260"
51 ATTACH HISTORY OF sec_row OF newsection TO secrow[ 153]
52
53 newsection-FIRST section IN sections WITH sec_row_number = 290
54 secrow[ 154] - NEW WINDOW CALLED "noderow290"
55 ATTACH HISTORY OF sec_row OF newsection TO secrow[ 154]
56
57 newsection-FIRST section IN sections WITH sec_row_number = 320
58 secrow[ 155] - NEW WINDOW CALLED "noderow320"
59 ATTACH HISTORY OF sec_row OF newsection TO secrow[ 155]
60
61 newsection-FIRST section IN sections WITH sec_row_number = 350
62 secrow[ 156] - NEW WINDOW CALLED "noderow350"
ATTACH HISTORY OF sec_row OF newsession TO secrow[156]

newsection-FIRST section IN sections WITH sec_row_number = 380
secrow[157]—NEW WINDOW CALLED "noderow380"
ATTACH HISTORY OF sec_row OF newsession TO secrow[157]

newsection-FIRST section IN sections WITH sec_row_number = 410
secrow[158]—NEW WINDOW CALLED "noderow410"
ATTACH HISTORY OF sec_row OF newsession TO secrow[158]

newsection-FIRST section IN sections WITH sec_row_number = 440
secrow[159]—NEW WINDOW CALLED "noderow440"
ATTACH HISTORY OF sec_row OF newsession TO secrow[159]

newsection-FIRST section IN sections WITH sec_row_number = 535
secrow[160]—NEW WINDOW CALLED "noderow535"
ATTACH HISTORY OF sec_row OF newsession TO secrow[160]

newsection-FIRST section IN sections WITH sec_row_number = 533
secrow[161]—NEW WINDOW CALLED "noderow533"
ATTACH HISTORY OF sec_row OF newsession TO secrow[161]

newsection-FIRST section IN sections WITH sec_row_number = 531
secrow[162]—NEW WINDOW CALLED "noderow531"
ATTACH HISTORY OF sec_row OF newsession TO secrow[162]

newsection-FIRST section IN sections WITH sec_row_number = 504
secrow[163]—NEW WINDOW CALLED "noderow504"
ATTACH HISTORY OF sec_row OF newsession TO secrow[163]

newsection-FIRST section IN sections WITH sec_row_number = 506
secrow[164]—NEW WINDOW CALLED "noderow506"
ATTACH HISTORY OF sec_row OF newsession TO secrow[164]

newsection-FIRST section IN sections WITH sec_row_number = 508
secrow[165]—NEW WINDOW CALLED "noderow508"
ATTACH HISTORY OF sec_row OF newsession TO secrow[165]

newsection-FIRST section IN sections WITH sec_row_number = 510
secrow[166]—NEW WINDOW CALLED "noderow510"
ATTACH HISTORY OF sec_row OF newsession TO secrow[166]

newsection-FIRST section IN sections WITH sec_row_number = 512
secrow[167]—NEW WINDOW CALLED "noderow512"
ATTACH HISTORY OF sec_row OF newsession TO secrow[167]

newsection-FIRST section IN sections WITH sec_row_number = 514
secrow[168]—NEW WINDOW CALLED "noderow514"
ATTACH HISTORY OF sec_row OF newsession TO secrow[168]

newsection-FIRST section IN sections WITH sec_row_number = 516
secrow[169]—NEW WINDOW CALLED "noderow516"
ATTACH HISTORY OF sec_row OF newsession TO secrow[169]

newsection-FIRST section IN sections WITH sec_row_number = 243
secrow[170]—NEW WINDOW CALLED "noderow243"
ATTACH HISTORY OF sec_row OF newsession TO secrow[170]

newsection-FIRST section IN sections WITH sec_row_number = 273
secrow[171]—NEW WINDOW CALLED "noderow273"
CREATING SECTIONS - page 3

125 ATTACH HISTORY OF sec_row OF newsection TO secrow(171)
126
127 newsection-FIRST section IN sections WITH sec_row_number = 303
128 secrow(172) +NEW WINDOW CALLED "noderow303"
129 ATTACH HISTORY OF sec_row OF newsection TO secrow(172)
130
131 newsection-FIRST section IN sections WITH sec_row_number = 333
132 secrow(173) +NEW WINDOW CALLED "noderow333"
133 ATTACH HISTORY OF sec_row OF newsection TO secrow(173)
134
135 newsection-FIRST section IN sections WITH sec_row_number = 363
136 secrow(174) +NEW WINDOW CALLED "noderow363"
137 ATTACH HISTORY OF sec_row OF newsection TO secrow(174)
138
139 newsection-FIRST section IN sections WITH sec_row_number = 393
140 secrow(175) +NEW WINDOW CALLED "noderow393"
141 ATTACH HISTORY OF sec_row OF newsection TO secrow(175)
142
143 newsection-FIRST section IN sections WITH sec_row_number = 423
144 secrow(176) +NEW WINDOW CALLED "noderow423"
145 ATTACH HISTORY OF sec_row OF newsection TO secrow(176)
146
147 newsection-FIRST section IN sections WITH sec_row_number = 453
148 secrow(177) +NEW WINDOW CALLED "noderow453"
149 ATTACH HISTORY OF sec_row OF newsection TO secrow(177)
150
151
152 @lekker met de stacks bezig joh zijn er niet zo veel
153
154 newstack-FIRST stack IN stacklist WITH stacknumber = c1
155 secrow(5) +NEW WINDOW CALLED "stack21"
156 ATTACH HISTORY OF stack_row OF newstack TO secrow(5)
157
158 newstack-FIRST stack IN stacklist WITH stacknumber = c2
159 secrow(6) +NEW WINDOW CALLED "stack22"
160 ATTACH HISTORY OF stack_row OF newstack TO secrow(6)
161
162 newstack-FIRST stack IN stacklist WITH stacknumber = c3
163 secrow(7) +NEW WINDOW CALLED "stack23"
164 ATTACH HISTORY OF stack_row OF newstack TO secrow(7)
165
166 newstack-FIRST stack IN stacklist WITH stacknumber = c4
167 secrow(8) +NEW WINDOW CALLED "stack24"
168 ATTACH HISTORY OF stack_row OF newstack TO secrow(8)
169
170 newstack-FIRST stack IN stacklist WITH stacknumber = c5
171 secrow(9) +NEW WINDOW CALLED "stack25"
172 ATTACH HISTORY OF stack_row OF newstack TO secrow(9)
173
174 newstack-FIRST stack IN stacklist WITH stacknumber = c6
175 secrow(10) +NEW WINDOW CALLED "stack26"
176 ATTACH HISTORY OF stack_row OF newstack TO secrow(10)
177
178 newstack-FIRST stack IN stacklist WITH stacknumber = c7
179 secrow(11) +NEW WINDOW CALLED "stack27"
180 ATTACH HISTORY OF stack_row OF newstack TO secrow(11)
181
182 newstack-FIRST stack IN stacklist WITH stacknumber = c8
183 secrow(12) +NEW WINDOW CALLED "stack28"
184 ATTACH HISTORY OF stack_row OF newstack TO secrow(12)
CREATING_SECTIONS - page 4

187 newstack-FIRST stack IN stacklist WITH stacknumber = c9
188 secrow[13] - NEW WINDOW CALLED "stack29"
189 ATTACH HISTORY OF stack_row OF newstack TO secrow[13]
190
191 newstack-FIRST stack IN stacklist WITH stacknumber = c10
192 secrow[14] - NEW WINDOW CALLED "stack30"
193 ATTACH HISTORY OF stack_row OF newstack TO secrow[14]
194
195 newstack-FIRST stack IN stacklist WITH stacknumber = 31
196 secrow[15] - NEW WINDOW CALLED "stack31"
197 ATTACH HISTORY OF stack_row OF newstack TO secrow[15]
198
199 newstack-FIRST stack IN stacklist WITH stacknumber = 32
200 secrow[16] - NEW WINDOW CALLED "stack32"
201 ATTACH HISTORY OF stack_row OF newstack TO secrow[16]
202
203 newstack-FIRST stack IN stacklist WITH stacknumber = 33
204 secrow[17] - NEW WINDOW CALLED "stack33"
205 ATTACH HISTORY OF stack_row OF newstack TO secrow[17]
206
207 newstack-FIRST stack IN stacklist WITH stacknumber = 34
208 secrow[18] - NEW WINDOW CALLED "stack34"
209 ATTACH HISTORY OF stack_row OF newstack TO secrow[18]
210
211 newstack-FIRST stack IN stacklist WITH stacknumber = 35
212 secrow[19] - NEW WINDOW CALLED "stack35"
213 ATTACH HISTORY OF stack_row OF newstack TO secrow[19]
214
215 newstack-FIRST stack IN stacklist WITH stacknumber = 36
216 secrow[20] - NEW WINDOW CALLED "stack36"
217 ATTACH HISTORY OF stack_row OF newstack TO secrow[20]
218
219 newstack-FIRST stack IN stacklist WITH stacknumber = 37
220 secrow[21] - NEW WINDOW CALLED "stack37"
221 ATTACH HISTORY OF stack_row OF newstack TO secrow[21]
222
223 newstack-FIRST stack IN stacklist WITH stacknumber = 38
224 secrow[22] - NEW WINDOW CALLED "stack38"
225 ATTACH HISTORY OF stack_row OF newstack TO secrow[22]
226
227 newstack-FIRST stack IN stacklist WITH stacknumber = 39
228 secrow[23] - NEW WINDOW CALLED "stack39"
229 ATTACH HISTORY OF stack_row OF newstack TO secrow[23]
230
231 newstack-FIRST stack IN stacklist WITH stacknumber = 40
232 secrow[24] - NEW WINDOW CALLED "stack40"
233 ATTACH HISTORY OF stack_row OF newstack TO secrow[24]
234
235 newstack-FIRST stack IN stacklist WITH stacknumber = 41
236 secrow[25] - NEW WINDOW CALLED "stack41"
237 ATTACH HISTORY OF stack_row OF newstack TO secrow[25]
238
239 newstack-FIRST stack IN stacklist WITH stacknumber = 42
240 secrow[26] - NEW WINDOW CALLED "stack42"
241 ATTACH HISTORY OF stack_row OF newstack TO secrow[26]
242
243 newstack-FIRST stack IN stacklist WITH stacknumber = 43
244 secrow[27] - NEW WINDOW CALLED "stack43"
245 ATTACH HISTORY OF stack_row OF newstack TO secrow[27]
246
247 newstack-FIRST stack IN stacklist WITH stacknumber = 44
248 secrow[28] - NEW WINDOW CALLED "stack44"
249 ATTACH HISTORY OF stack_row OF newstack TO secrow[28]
250
251 newstack-FIRST stack IN stacklist WITH stacknumber = 45
252 secrow[29] → NEW WINDOW CALLED "stack45"
253 ATTACH HISTORY OF stack_row OF newstack TO secrow[29]
254
255 newstack-FIRST stack IN stacklist WITH stacknumber = 46
256 secrow[30] → NEW WINDOW CALLED "stack46"
257 ATTACH HISTORY OF stack_row OF newstack TO secrow[30]
258
259 newstack-FIRST stack IN stacklist WITH stacknumber = 47
260 secrow[31] → NEW WINDOW CALLED "stack47"
261 ATTACH HISTORY OF stack_row OF newstack TO secrow[31]
262
263 newstack-FIRST stack IN stacklist WITH stacknumber = 48
264 secrow[32] → NEW WINDOW CALLED "stack48"
265 ATTACH HISTORY OF stack_row OF newstack TO secrow[32]
266
267 newstack-FIRST stack IN stacklist WITH stacknumber = 49
268 secrow[33] → NEW WINDOW CALLED "stack49"
269 ATTACH HISTORY OF stack_row OF newstack TO secrow[33]
270
271 newstack-FIRST stack IN stacklist WITH stacknumber = 50
272 secrow[34] → NEW WINDOW CALLED "stack50"
273 ATTACH HISTORY OF stack_row OF newstack TO secrow[34]
274
275 newstack-FIRST stack IN stacklist WITH stacknumber = 51
276 secrow[35] → NEW WINDOW CALLED "stack51"
277 ATTACH HISTORY OF stack_row OF newstack TO secrow[35]
278
279 newstack-FIRST stack IN stacklist WITH stacknumber = 52
280 secrow[36] → NEW WINDOW CALLED "stack52"
281 ATTACH HISTORY OF stack_row OF newstack TO secrow[36]
282
283 newstack-FIRST stack IN stacklist WITH stacknumber = 53
284 secrow[37] → NEW WINDOW CALLED "stack53"
285 ATTACH HISTORY OF stack_row OF newstack TO secrow[37]
286
287 newstack-FIRST stack IN stacklist WITH stacknumber = 54
288 secrow[38] → NEW WINDOW CALLED "stack54"
289 ATTACH HISTORY OF stack_row OF newstack TO secrow[38]
290
291 newstack-FIRST stack IN stacklist WITH stacknumber = 55
292 secrow[39] → NEW WINDOW CALLED "stack55"
293 ATTACH HISTORY OF stack_row OF newstack TO secrow[39]
294
295 newstack-FIRST stack IN stacklist WITH stacknumber = 56
296 secrow[40] → NEW WINDOW CALLED "stack56"
297 ATTACH HISTORY OF stack_row OF newstack TO secrow[40]
298
299 newstack-FIRST stack IN stacklist WITH stacknumber = 57
300 secrow[41] → NEW WINDOW CALLED "stack57"
301 ATTACH HISTORY OF stack_row OF newstack TO secrow[41]
302
303 newstack-FIRST stack IN stacklist WITH stacknumber = 58
304 secrow[42] → NEW WINDOW CALLED "stack58"
305 ATTACH HISTORY OF stack_row OF newstack TO secrow[42]
306
307 newstack-FIRST stack IN stacklist WITH stacknumber = 59
308 secrow[43] → NEW WINDOW CALLED "stack59"
309 ATTACH HISTORY OF stack_row OF newstack TO secrow[43]
310
newstack-FIRST stack IN stacklist WITH stacknumber = 60
secrow[ 44] → NEW WINDOW CALLED "stack60"
ATTACH HISTORY OF stack_row OF newstack TO secrow[ 44]

newstack-FIRST stack IN stacklist WITH stacknumber = 61
secrow[ 45] → NEW WINDOW CALLED "stack61"
ATTACH HISTORY OF stack_row OF newstack TO secrow[ 45]

newstack-FIRST stack IN stacklist WITH stacknumber = 62
secrow[ 46] → NEW WINDOW CALLED "stack62"
ATTACH HISTORY OF stack_row OF newstack TO secrow[ 46]

newstack-FIRST stack IN stacklist WITH stacknumber = 63
secrow[ 47] → NEW WINDOW CALLED "stack63"
ATTACH HISTORY OF stack_row OF newstack TO secrow[ 47]

newstack-FIRST stack IN stacklist WITH stacknumber = 64
secrow[ 48] → NEW WINDOW CALLED "stack64"
ATTACH HISTORY OF stack_row OF newstack TO secrow[ 48]

newstack-FIRST stack IN stacklist WITH stacknumber = 65
secrow[ 49] → NEW WINDOW CALLED "stack65"
ATTACH HISTORY OF stack_row OF newstack TO secrow[ 49]

newstack-FIRST stack IN stacklist WITH stacknumber = 66
secrow[ 50] → NEW WINDOW CALLED "stack66"
ATTACH HISTORY OF stack_row OF newstack TO secrow[ 50]

newstack-FIRST stack IN stacklist WITH stacknumber = 67
secrow[ 51] → NEW WINDOW CALLED "stack67"
ATTACH HISTORY OF stack_row OF newstack TO secrow[ 51]

newstack-FIRST stack IN stacklist WITH stacknumber = 68
secrow[ 52] → NEW WINDOW CALLED "stack68"
ATTACH HISTORY OF stack_row OF newstack TO secrow[ 52]

newstack-FIRST stack IN stacklist WITH stacknumber = 69
secrow[ 53] → NEW WINDOW CALLED "stack69"
ATTACH HISTORY OF stack_row OF newstack TO secrow[ 53]

newstack-FIRST stack IN stacklist WITH stacknumber = 70
secrow[ 54] → NEW WINDOW CALLED "stack70"
ATTACH HISTORY OF stack_row OF newstack TO secrow[ 54]

newstack-FIRST stack IN stacklist WITH stacknumber = 71
secrow[ 55] → NEW WINDOW CALLED "stack71"
ATTACH HISTORY OF stack_row OF newstack TO secrow[ 55]

newstack-FIRST stack IN stacklist WITH stacknumber = 72
secrow[ 56] → NEW WINDOW CALLED "stack72"
ATTACH HISTORY OF stack_row OF newstack TO secrow[ 56]

newstack-FIRST stack IN stacklist WITH stacknumber = 73
secrow[ 57] → NEW WINDOW CALLED "stack73"
ATTACH HISTORY OF stack_row OF newstack TO secrow[ 57]

newstack-FIRST stack IN stacklist WITH stacknumber = 74
secrow[ 58] → NEW WINDOW CALLED "stack74"
ATTACH HISTORY OF stack_row OF newstack TO secrow[ 58]

newstack-FIRST stack IN stacklist WITH stacknumber = 75
secrow[ 59] → NEW WINDOW CALLED "stack75"
ATTACH HISTORY OF stack_row OF newstack TO secrow[59]
newstack-FIRST stack IN stacklist WITH stacknumber = 76
secrow[60] → NEW WINDOW CALLED "stack76"
ATTACH HISTORY OF stack_row OF newstack TO secrow[60]
newstack-FIRST stack IN stacklist WITH stacknumber = 77
secrow[61] → NEW WINDOW CALLED "stack77"
ATTACH HISTORY OF stack_row OF newstack TO secrow[61]
newstack-FIRST stack IN stacklist WITH stacknumber = 78
secrow[62] → NEW WINDOW CALLED "stack78"
ATTACH HISTORY OF stack_row OF newstack TO secrow[62]
newstack-FIRST stack IN stacklist WITH stacknumber = 79
secrow[63] → NEW WINDOW CALLED "stack79"
ATTACH HISTORY OF stack_row OF newstack TO secrow[63]
newstack-FIRST stack IN stacklist WITH stacknumber = 80
secrow[64] → NEW WINDOW CALLED "stack80"
ATTACH HISTORY OF stack_row OF newstack TO secrow[64]
newstack-FIRST stack IN stacklist WITH stacknumber = 81
secrow[65] → NEW WINDOW CALLED "stack81"
ATTACH HISTORY OF stack_row OF newstack TO secrow[65]
newstack-FIRST stack IN stacklist WITH stacknumber = 82
secrow[66] → NEW WINDOW CALLED "stack82"
ATTACH HISTORY OF stack_row OF newstack TO secrow[66]
newstack-FIRST stack IN stacklist WITH stacknumber = 83
secrow[67] → NEW WINDOW CALLED "stack83"
ATTACH HISTORY OF stack_row OF newstack TO secrow[67]
newstack-FIRST stack IN stacklist WITH stacknumber = 84
secrow[68] → NEW WINDOW CALLED "stack84"
ATTACH HISTORY OF stack_row OF newstack TO secrow[68]
newstack-FIRST stack IN stacklist WITH stacknumber = 85
secrow[69] → NEW WINDOW CALLED "stack85"
ATTACH HISTORY OF stack_row OF newstack TO secrow[69]
newstack-FIRST stack IN stacklist WITH stacknumber = 86
secrow[70] → NEW WINDOW CALLED "stack86"
ATTACH HISTORY OF stack_row OF newstack TO secrow[70]
newstack-FIRST stack IN stacklist WITH stacknumber = 87
secrow[71] → NEW WINDOW CALLED "stack87"
ATTACH HISTORY OF stack_row OF newstack TO secrow[71]
newstack-FIRST stack IN stacklist WITH stacknumber = 88
secrow[72] → NEW WINDOW CALLED "stack88"
ATTACH HISTORY OF stack_row OF newstack TO secrow[72]
newstack-FIRST stack IN stacklist WITH stacknumber = 89
secrow[73] → NEW WINDOW CALLED "stack89"
ATTACH HISTORY OF stack_row OF newstack TO secrow[73]
newstack-FIRST stack IN stacklist WITH stacknumber = 90
secrow[74] → NEW WINDOW CALLED "stack90"
ATTACH HISTORY OF stack_row OF newstack TO secrow[74]
newstack-FIRST stack IN stacklist WITH stacknumber = 122
secrow[106]→NEW WINDOW CALLED "stack122"
ATTACH HISTORY OF stack_row OF newstack TO secrow[106]

newstack-FIRST stack IN stacklist WITH stacknumber = 123
secrow[107]→NEW WINDOW CALLED "stack123"
ATTACH HISTORY OF stack_row OF newstack TO secrow[107]

newstack-FIRST stack IN stacklist WITH stacknumber = 124
secrow[108]→NEW WINDOW CALLED "stack124"
ATTACH HISTORY OF stack_row OF newstack TO secrow[108]

newstack-FIRST stack IN stacklist WITH stacknumber = 125
secrow[109]→NEW WINDOW CALLED "stack125"
ATTACH HISTORY OF stack_row OF newstack TO secrow[109]

newstack-FIRST stack IN stacklist WITH stacknumber = 126
secrow[110]→NEW WINDOW CALLED "stack126"
ATTACH HISTORY OF stack_row OF newstack TO secrow[110]

newstack-FIRST stack IN stacklist WITH stacknumber = 127
secrow[111]→NEW WINDOW CALLED "stack127"
ATTACH HISTORY OF stack_row OF newstack TO secrow[111]

newstack-FIRST stack IN stacklist WITH stacknumber = 128
secrow[112]→NEW WINDOW CALLED "stack128"
ATTACH HISTORY OF stack_row OF newstack TO secrow[112]

newstack-FIRST stack IN stacklist WITH stacknumber = 129
secrow[113]→NEW WINDOW CALLED "stack129"
ATTACH HISTORY OF stack_row OF newstack TO secrow[113]

newstack-FIRST stack IN stacklist WITH stacknumber = 130
secrow[114]→NEW WINDOW CALLED "stack130"
ATTACH HISTORY OF stack_row OF newstack TO secrow[114]

newstack-FIRST stack IN stacklist WITH stacknumber = 131
secrow[115]→NEW WINDOW CALLED "stack131"
ATTACH HISTORY OF stack_row OF newstack TO secrow[115]

newstack-FIRST stack IN stacklist WITH stacknumber = 132
secrow[116]→NEW WINDOW CALLED "stack132"
ATTACH HISTORY OF stack_row OF newstack TO secrow[116]

newstack-FIRST stack IN stacklist WITH stacknumber = 133
secrow[117]→NEW WINDOW CALLED "stack133"
ATTACH HISTORY OF stack_row OF newstack TO secrow[117]

newstack-FIRST stack IN stacklist WITH stacknumber = 134
secrow[118]→NEW WINDOW CALLED "stack134"
ATTACH HISTORY OF stack_row OF newstack TO secrow[118]

newstack-FIRST stack IN stacklist WITH stacknumber = 135
secrow[119]→NEW WINDOW CALLED "stack135"
ATTACH HISTORY OF stack_row OF newstack TO secrow[119]

newstack-FIRST stack IN stacklist WITH stacknumber = 136
secrow[120]→NEW WINDOW CALLED "stack136"
ATTACH HISTORY OF stack_row OF newstack TO secrow[120]

newstack-FIRST stack IN stacklist WITH stacknumber = 137
secrow[121]→NEW WINDOW CALLED "stack137"
ATTACH HISTORY OF stack_row OF newstack TO secrow[121]
newstack-FIRST stack IN stacklist WITH stacknumber = 138
secrow[122]-NEW WINDOW CALLED "stack138"
ATTACH HISTORY OF stack_row OF newstack TO secrow[122]
newstack-FIRST stack IN stacklist WITH stacknumber = 139
secrow[123]-NEW WINDOW CALLED "stack139"
ATTACH HISTORY OF stack_row OF newstack TO secrow[123]
newstack-FIRST stack IN stacklist WITH stacknumber = 140
secrow[124]-NEW WINDOW CALLED "stack140"
ATTACH HISTORY OF stack_row OF newstack TO secrow[124]
newstack-FIRST stack IN stacklist WITH stacknumber = 141
secrow[125]-NEW WINDOW CALLED "stack141"
ATTACH HISTORY OF stack_row OF newstack TO secrow[125]
newstack-FIRST stack IN stacklist WITH stacknumber = 142
secrow[126]-NEW WINDOW CALLED "stack142"
ATTACH HISTORY OF stack_row OF newstack TO secrow[126]
newstack-FIRST stack IN stacklist WITH stacknumber = 143
secrow[127]-NEW WINDOW CALLED "stack143"
ATTACH HISTORY OF stack_row OF newstack TO secrow[127]
newstack-FIRST stack IN stacklist WITH stacknumber = 144
secrow[128]-NEW WINDOW CALLED "stack144"
ATTACH HISTORY OF stack_row OF newstack TO secrow[128]
newstack-FIRST stack IN stacklist WITH stacknumber = 145
secrow[129]-NEW WINDOW CALLED "stack145"
ATTACH HISTORY OF stack_row OF newstack TO secrow[129]
newstack-FIRST stack IN stacklist WITH stacknumber = 146
secrow[130]-NEW WINDOW CALLED "stack146"
ATTACH HISTORY OF stack_row OF newstack TO secrow[130]
newstack-FIRST stack IN stacklist WITH stacknumber = 147
secrow[131]-NEW WINDOW CALLED "stack147"
ATTACH HISTORY OF stack_row OF newstack TO secrow[131]
newstack-FIRST stack IN stacklist WITH stacknumber = 148
secrow[132]-NEW WINDOW CALLED "stack148"
ATTACH HISTORY OF stack_row OF newstack TO secrow[132]
newstack-FIRST stack IN stacklist WITH stacknumber = 149
secrow[133]-NEW WINDOW CALLED "stack149"
ATTACH HISTORY OF stack_row OF newstack TO secrow[133]
newstack-FIRST stack IN stacklist WITH stacknumber = 150
secrow[134]-NEW WINDOW CALLED "stack150"
ATTACH HISTORY OF stack_row OF newstack TO secrow[134]
newstack-FIRST stack IN stacklist WITH stacknumber = 151
secrow[135]-NEW WINDOW CALLED "stack151"
ATTACH HISTORY OF stack_row OF newstack TO secrow[135]
newstack-FIRST stack IN stacklist WITH stacknumber = 152
secrow[136]-NEW WINDOW CALLED "stack152"
ATTACH HISTORY OF stack_row OF newstack TO secrow[136]
683 newstack-FIRST stack IN stacklist WITH stacknumber = 153
684 secrow[137]-NEW WINDOW CALLED "stack153"
685 ATTACH HISTORY OF stack_row OF newstack TO secrow[137]
686
687 newstack-FIRST stack IN stacklist WITH stacknumber = 154
688 secrow[138]-NEW WINDOW CALLED "stack154"
689 ATTACH HISTORY OF stack_row OF newstack TO secrow[138]
690
691 newstack-FIRST stack IN stacklist WITH stacknumber = 155
692 secrow[139]-NEW WINDOW CALLED "stack155"
693 ATTACH HISTORY OF stack_row OF newstack TO secrow[139]
694
695 newstack-FIRST stack IN stacklist WITH stacknumber = 156
696 secrow[140]-NEW WINDOW CALLED "stack156"
697 ATTACH HISTORY OF stack_row OF newstack TO secrow[140]
698
699 newstack-FIRST stack IN stacklist WITH stacknumber = 157
700 secrow[141]-NEW WINDOW CALLED "stack157"
701 ATTACH HISTORY OF stack_row OF newstack TO secrow[141]
702
703 newstack-FIRST stack IN stacklist WITH stacknumber = 158
704 secrow[142]-NEW WINDOW CALLED "stack158"
705 ATTACH HISTORY OF stack_row OF newstack TO secrow[142]
706
707 newstack-FIRST stack IN stacklist WITH stacknumber = 159
708 secrow[143]-NEW WINDOW CALLED "stack159"
709 ATTACH HISTORY OF stack_row OF newstack TO secrow[143]
710
711 newstack-FIRST stack IN stacklist WITH stacknumber = 160
712 secrow[144]-NEW WINDOW CALLED "stack160"
713 ATTACH HISTORY OF stack_row OF newstack TO secrow[144]
714 RETURN
create portout is a macro that creates portainer outportations
1 newportloc = FIRST portloc IN portlocations WITH portlocnumber = 3
2 port3queue ← NEW WINDOW CALLED "port3queue"
3 ATTACH HISTORY OF portlocqueue OF newportloc TO port3queue
4 newportloc = FIRST portloc IN portlocations WITH portlocnumber = 4
5 port4queue ← NEW WINDOW CALLED "port4queue"
6 ATTACH HISTORY OF portlocqueue OF newportloc TO port4queue
7 newportloc = FIRST portloc IN portlocations WITH portlocnumber = 5
8 port5queue ← NEW WINDOW CALLED "port5queue"
9 ATTACH HISTORY OF portlocqueue OF newportloc TO port5queue
10 newportloc = FIRST portloc IN portlocations WITH portlocnumber = 6
11 port6queue ← NEW WINDOW CALLED "port6queue"
12 ATTACH HISTORY OF portlocqueue OF newportloc TO port6queue
13 newportloc = FIRST portloc IN portlocations WITH portlocnumber = 7
14 port7queue ← NEW WINDOW CALLED "port7queue"
15 ATTACH HISTORY OF portlocqueue OF newportloc TO port7queue
16 newportloc = FIRST portloc IN portlocations WITH portlocnumber = 8
17 port8queue ← NEW WINDOW CALLED "port8queue"
18 ATTACH HISTORY OF portlocqueue OF newportloc TO port8queue
19 newportloc = FIRST portloc IN portlocations WITH portlocnumber = 9
20 port9queue ← NEW WINDOW CALLED "port9queue"
21 ATTACH HISTORY OF portlocqueue OF newportloc TO port9queue
22 newportloc = FIRST portloc IN portlocations WITH portlocnumber = 10
23 port10queue ← NEW WINDOW CALLED "port10queue"
24 ATTACH HISTORY OF portlocqueue OF newportloc TO port10queue
25 newportloc = FIRST portloc IN portlocations WITH portlocnumber = 11
26 port11queue ← NEW WINDOW CALLED "port11queue"
27 ATTACH HISTORY OF portlocqueue OF newportloc TO port11queue
28 newportloc = FIRST portloc IN portlocations WITH portlocnumber = 12
29 port12queue ← NEW WINDOW CALLED "port12queue"
30 ATTACH HISTORY OF portlocqueue OF newportloc TO port12queue
31 newportloc = FIRST portloc IN portlocations WITH portlocnumber = 13
32 port13queue ← NEW WINDOW CALLED "port13queue"
33 ATTACH HISTORY OF portlocqueue OF newportloc TO port13queue
34 newportloc = FIRST portloc IN portlocations WITH portlocnumber = 14
35 port14queue ← NEW WINDOW CALLED "port14queue"
36 ATTACH HISTORY OF portlocqueue OF newportloc TO port14queue
37 newportloc = FIRST portloc IN portlocations WITH portlocnumber = 15
38 port15queue ← NEW WINDOW CALLED "port15queue"
39 ATTACH HISTORY OF portlocqueue OF newportloc TO port15queue
40 newportloc = FIRST portloc IN portlocations WITH portlocnumber = 16
41 port16queue ← NEW WINDOW CALLED "port16queue"
42 ATTACH HISTORY OF portlocqueue OF newportloc TO port16queue
43 RETURN
portainertimes is the macro that creates the portainer cycletimes
there are two different cycletimes dependent of the size of the vessel (feeder mega vessel)
And there is a difference between the loading and the unloading of a vessel

qq is the time the portainer eventually has to work
PORTAINERTIMES - page 1

1 @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

2 @Deze formule stelt wel eisen aan het aantal containers per schip hier -  
3 @moet wel rekening mee gehouden worden!!!  

4 @deze formule wordt gebruikt bij het lossen van de schepen  

5 IF big OF ptship = FALSE & ptship BELONGS TO unloadingship  
6 n ← LENGTH OF portainerperberth[ shipberthenumber OF ptship]  
7 PER-ABS(totalunload OF ptship - total OF ptship)  
8 pers← (total OF ptship/3)  
9 rep-FLOOR(totalunload OF ptship/pers)  
10 QQ ← (PER)MODULO(12*n)*(23/((12*n)-1))+ 28.5+(pers-ABS-(  
11 totalunload OF ptship-pers*rep)))*(16/pers)  
12 END  
13 IF BIG OF PTSHIP = TRUE & ptship BELONGS TO unloadingship  
14 n ← LENGTH OF portainerperberth[ shipberthenumber OF ptship]  
15 PER-ABS(totalunload OF ptship - total OF ptship)  
16 pers← (total OF ptship/3)  
17 rep-FLOOR(totalunload OF ptship/pers)  
18 QQ ← (PER)MODULO(18*n)*(23/((18*n)-1))+ 28.5+(pers-ABS-(  
19 totalunload OF ptship-pers*rep)))*(16/pers)  
20 END  
21 @deze formule wordt gebruikt bij het laden van de schepen  
22  
23 IF big OF ptship = FALSE & ptship BELONGS TO loadingship  
24 n ← LENGTH OF portainerperberth[ shipberthenumber OF ptship]  
25 PER-ABS(totalload OF ptship - totalu OF ptship)  
26 pers← (totalu OF ptship/3)  
27 rep-FLOOR(totalload OF ptship/pers)  
28 QQ ← ((12*n-PER)MODULO(12*n))*(23/((12*n)-1))→  
29 + 28.5+(totalload OF ptship-pers*rep)))*(16/pers)  
30 END  
31 IF BIG OF PTSHIP = TRUE & ptship BELONGS TO loadingship  
32 n ← LENGTH OF portainerperberth[ shipberthenumber OF ptship]  
33 PER-ABS(totalload OF ptship - totalu OF ptship)  
34 pers← (totalu OF ptship/3)  
35 rep-FLOOR(totalload OF ptship/pers)  
36 QQ ← (18*n-PER)MODULO(18*n)*(23/((18*n)-1)))→  
37 + 28.5+(totalload OF ptship-pers*rep)))*(16/pers)  
38 END  
39 RETURN
Appendix 28-IV Clarifying word list
clarifying word list

A  Aantalagv is an attribute of the crane of the type integer indicating the number of containers that are dropped at a service point by a rgv unloading a vessel, these containers still have to be put into the stack by the stacking crane
  y creation is a macro that chooses the cranes that have to serve the stacks. With this macro the number of cranes in the stacks can be varied
  acc is the acceleration of the rgv's used during the run
  Activeshiplist a set containing the active vessels at the terminal
  actlocnr is the actual location number of a portainer

  Agv_berth_stack is a macro that creates the route from a berth to the stack
  agv_unload creates the route for the rgv to get to the stack from the berth
  agv_unload_back is a macro determining the route the rgv has to take from the yard to the berth
  agv_stack_berth is a macro that creates the route to get from any stack to a berth
  agv_stack_stack is a macro that creates the route from any stack to another stack
  agv_uneasy_b_s is a macro that creates a route from the berth to a stack in the transition between loading a vessel and shuffling

  Agvberthshufflelist [1 to 3] are 3 sets containing all the rgv's shuffling from that berth
  Agvparking a set containing the not active rgv's
  agvppactive is a set containing the number of agv's (rgv's) per portainer active
  agvppactive_odd is a set containing the number of odd agv's per portainer active
  Agwworklist a set containing rgv's with work requests for the trafficmaster
  Anchorag is a set containing the vessels waiting to berth
  assignedqueue is a set containing rgv's assigned to the crane.

B  Big is an attribute of ship of the type logical, if big is "false" it means this is a feeder
  Bye is an attribute of the rgv of the type logical. If bye of a rgv is true it has to go to the parking

C  claimedlength is the length the rgv needs to enter the main track
  claimset is a set of nodes that have to be claimed in order to enter the main track
  control box is the box that appears when starting the run this enables the user to check some graphs after the run
  conversation is the macro that creates the control box
  crane cycletime is a macro that creates the cycletimes for the stacking cranes
  crane outportations is a macro that creates the outportations for the stacking cranes
  crane_cycletime is a macro which creates a cycle times for the cranes
  Cranelist a set containing all the cranes

  cranequeue is a set containing the rgv's waiting to be served by the crane
  Create outportations is a macro that creates the outportations
  create portout is a macro that creates all the portainer outportations
  create platform is a macro creating the platforms
  create_stack is the macro that creates the stacks this macro is called upon by the main
  creating sections is a macro that attaches existing queues of sections to windows. In order to
  monitor these queues. In this way node rows and stack rows can be monitored.

D  distance is the minimum distance between rgv's when driving on the terminal

E  easy loading concerns the loading where the containers that have to be loaded are already located in the stacks behind the right berth
  Easyload is an attribute of a rgv referring to the easy loading method of a vessel.
  end point is a reference to the last node of a section
  Entrance of the rgv is a set containing the starting point of a route of the rgv
  Even rgv's take the bottom route when driving at the terminal
  exit the first node where the rgv has to leave the main track
  exittime is a specific time for a rgv to pull up and leave a node

  Exportgen is the exportgenerator and generates the export containers for the vessels
firstshuffle is an attribute of the rgv and indicates this rgv is shuffling for the first time
fromtob[1,2] is an attribute telling how many containers at berth 1 are destined to go to berth 2
full_speed is the top speed of the rgv, this speed can be reached on the main tracks
futlocnr is the future location number of the portainer. The new location of the portainer.
Goneshiplist a set containing the terminated vessels, in this way their attributes can be saved
im40 of a vessel is the number of 40 feet import containers
Importgenerator generates the import containers for the vessels
inplacecounter is the counter that counts the containers that are inplace behind that specific berth
Klaarcount is the counter that counts containers that are placed on the service points in a stack
ready to be taken away by a rgv loading a vessel
Lodedagv is a logical, if false it means the rgv is not carrying a container
Loadingship a set containing all the loading vessels at the terminal
Loadyardallocation is a macro that chooses where the rgv has to get a load container
lowspeed is the speed of the rgv in the stacks and at the short cuts
n_in is a reference to a node, the entrance of the stack
nd_x is the x co-ordinate of a node
nd_y is the y co-ordinate of a node
node_distance refers to the safety distance at the tracks, this can differ per track
Node_list is a set containing all the nodes
nodespeed id the speed of a rgv when it crosses a node
occupied[1] is a logical if occupied [1] is true it means berth1 is occupied by a vessel
Odd rgv's take the top route at the terminal when they want to get somewhere at the terminal
Outportation creates the output files
Owstack is a stack behind the portainer the rgv was working for in the beginning
Pal is portainer active list
Platforms is a set containing all the platforms
Portaineractivelist a set containing all the active portainers
Portainerperberth [1 to 3] keeps count of the active portainers per berth
Portainertimes is the macro that creates the portainer cycle times
Portasqueue is a set containing rgv's assigned to that portlocation.
Portloc is short for portainerlocation
Portlocations a set containing all the portlocations
Portlocqueue is a set containing rgv's waiting under the portainer to be served
Preclaimnodes is a macro called upon by the agv responsible for claiming nodes when more
than one node needs to be claimed
Quaylocation[l] is a logical if true it means this location is taken by a portainer
Quayrall a set containing all the portainers
Route of the rgv is a set containing all the exits from the main route the rgv has to take to reach
its destination
S_length is the length of a section calculated by the function distance to
sbnr is the stack berth number and identifies behind which berth the stacks are located where the
agv is shuffling for if sbnr =0 the rgv is working in the stacks behind its original portlocation (x)
sections is a set containing all the sections
shipanchoragetime is an attribute of the vessel referring to the time in berth
st_x is the actual x co-ordinate of the stack on the terminal
st_y is the actual y co-ordinate of the stack on the terminal
stack row a set containing all the rgv's that are in the stack at that moment
stackcount are the reserved service points in the stack
Stacklist a set containing all the stacks
superShortcut is a macro designed for an exception at the terminal.
tex is the total of containers to be loaded unto the vessel
Timebig refers to the interarrivaltime of a mega vessel
Timesmall refers to the interarrivaltime of a feeder
**timportloc** is the portlocation the rgv is working for

**tmres40lim** is the percentage 40 feet import containers of all the import containers

**tmrestotlim** is the percentage import containers still to be unloaded of the whole

**tmtotalex** is the total of containers to be loaded unto the vessel - total already loaded

**tmtotalim** is the total of containers left on the vessel that have to be unloaded

**totalload** is the number of containers the vessel has to receive at the terminal here

**totalu** is the start value of totalload, this value doesn't change in contrast to totalload

**U**  
Unloadinship a set containing all the vessels that are unloading

**unloadyardallocation** is a macro that creates the stacks for the rgv's to deliver

the unload containers

**W**  
**way_in_stack** is a attribute of the stack referring to the way in a stack

**X**  
**x** is an attribute of the rgv referring to a portainerlocation

**y** refers to the current portlocation of the rgv

**Y**  
**y_creation** is a macro responsible for determining which stacking crane works in which stack

**yardshuffleallocation** is the macro that creates the locations for the rgv to retrieve

containers from when shuffling
### Appendix 27-5. Input file for the lay out of the terminal

@These sets of 19 represent main tracks, the inter berth lanes @
@node number, x co-ordinate and y co-ordinate @

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>8</td>
<td>194</td>
<td>@in the final version of the program 'eindversie' this maintrack is @</td>
</tr>
<tr>
<td>2</td>
<td>70</td>
<td>194</td>
<td>@ not used. @</td>
</tr>
<tr>
<td>3</td>
<td>132</td>
<td>194</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>194</td>
<td>194</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>256</td>
<td>194</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>318</td>
<td>194</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>380</td>
<td>194</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>442</td>
<td>194</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>504</td>
<td>194</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>566</td>
<td>194</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>628</td>
<td>194</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>690</td>
<td>194</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>752</td>
<td>194</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>814</td>
<td>194</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>876</td>
<td>194</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>938</td>
<td>194</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>1000</td>
<td>194</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>1062</td>
<td>194</td>
<td>@in the final version of the program 'eindversie' this maintrack is @</td>
</tr>
<tr>
<td>19</td>
<td>1124</td>
<td>194</td>
<td>@ not used. @</td>
</tr>
<tr>
<td></td>
<td>-1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| 20| 1124| 186  | @in the final version of the program 'eindversie' this maintrack is @ |
| 21| 1062| 186  | @ not used. @               |
| 22| 1000| 186  |                              |
| 23| 938 | 186  |                              |
| 24| 876 | 186  |                              |
| 25| 814 | 186  |                              |
| 26| 752 | 186  |                              |
| 27| 690 | 186  |                              |
| 28| 628 | 186  |                              |
| 29| 566 | 186  |                              |
| 30| 504 | 186  |                              |
| 31| 442 | 186  |                              |
| 32| 380 | 186  |                              |
| 33| 318 | 186  |                              |
| 34| 256 | 186  |                              |
| 35| 194 | 186  |                              |
| 36| 132 | 186  |                              |
| 37| 70  | 186  |                              |
| 38| 8   | 186  |                              |
|   | -1 |      |                              |

| 39| 8   | 178  |                              |
| 40| 70  | 178  |                              |
| 41| 132 | 178  |                              |
| 42| 194 | 178  |                              |
| 43| 256 | 178  |                              |
| 44| 318 | 178  |                              |
| 45| 380 | 178  |                              |
| 46| 442 | 178  |                              |

1
<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>49</td>
<td>504</td>
<td>178</td>
<td></td>
</tr>
<tr>
<td>50</td>
<td>566</td>
<td>178</td>
<td></td>
</tr>
<tr>
<td>51</td>
<td>628</td>
<td>178</td>
<td></td>
</tr>
<tr>
<td>52</td>
<td>690</td>
<td>178</td>
<td></td>
</tr>
<tr>
<td>53</td>
<td>752</td>
<td>178</td>
<td></td>
</tr>
<tr>
<td>54</td>
<td>814</td>
<td>178</td>
<td></td>
</tr>
<tr>
<td>55</td>
<td>876</td>
<td>178</td>
<td></td>
</tr>
<tr>
<td>56</td>
<td>938</td>
<td>178</td>
<td></td>
</tr>
<tr>
<td>57</td>
<td>1000</td>
<td>178</td>
<td></td>
</tr>
<tr>
<td>58</td>
<td>1062</td>
<td>178</td>
<td></td>
</tr>
<tr>
<td>59</td>
<td>1124</td>
<td>178</td>
<td></td>
</tr>
<tr>
<td>-1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>60</td>
<td>1124</td>
<td>170</td>
<td></td>
</tr>
<tr>
<td>61</td>
<td>1062</td>
<td>170</td>
<td></td>
</tr>
<tr>
<td>62</td>
<td>1000</td>
<td>170</td>
<td></td>
</tr>
<tr>
<td>63</td>
<td>938</td>
<td>170</td>
<td></td>
</tr>
<tr>
<td>64</td>
<td>876</td>
<td>170</td>
<td></td>
</tr>
<tr>
<td>65</td>
<td>814</td>
<td>170</td>
<td></td>
</tr>
<tr>
<td>66</td>
<td>752</td>
<td>170</td>
<td></td>
</tr>
<tr>
<td>67</td>
<td>690</td>
<td>170</td>
<td></td>
</tr>
<tr>
<td>68</td>
<td>628</td>
<td>170</td>
<td></td>
</tr>
<tr>
<td>69</td>
<td>566</td>
<td>170</td>
<td></td>
</tr>
<tr>
<td>70</td>
<td>504</td>
<td>170</td>
<td></td>
</tr>
<tr>
<td>71</td>
<td>442</td>
<td>170</td>
<td></td>
</tr>
<tr>
<td>72</td>
<td>380</td>
<td>170</td>
<td></td>
</tr>
<tr>
<td>73</td>
<td>318</td>
<td>170</td>
<td></td>
</tr>
<tr>
<td>74</td>
<td>256</td>
<td>170</td>
<td></td>
</tr>
<tr>
<td>75</td>
<td>194</td>
<td>170</td>
<td></td>
</tr>
<tr>
<td>76</td>
<td>132</td>
<td>170</td>
<td></td>
</tr>
<tr>
<td>77</td>
<td>70</td>
<td>170</td>
<td></td>
</tr>
<tr>
<td>78</td>
<td>8</td>
<td>170</td>
<td></td>
</tr>
<tr>
<td>-1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>81</td>
<td>8</td>
<td>162</td>
<td></td>
</tr>
<tr>
<td>82</td>
<td>70</td>
<td>162</td>
<td></td>
</tr>
<tr>
<td>83</td>
<td>132</td>
<td>162</td>
<td></td>
</tr>
<tr>
<td>84</td>
<td>194</td>
<td>162</td>
<td></td>
</tr>
<tr>
<td>85</td>
<td>256</td>
<td>162</td>
<td></td>
</tr>
<tr>
<td>86</td>
<td>318</td>
<td>162</td>
<td></td>
</tr>
<tr>
<td>87</td>
<td>380</td>
<td>162</td>
<td></td>
</tr>
<tr>
<td>88</td>
<td>442</td>
<td>162</td>
<td></td>
</tr>
<tr>
<td>89</td>
<td>504</td>
<td>162</td>
<td></td>
</tr>
<tr>
<td>90</td>
<td>566</td>
<td>162</td>
<td></td>
</tr>
<tr>
<td>91</td>
<td>628</td>
<td>162</td>
<td></td>
</tr>
<tr>
<td>92</td>
<td>690</td>
<td>162</td>
<td></td>
</tr>
<tr>
<td>93</td>
<td>752</td>
<td>162</td>
<td></td>
</tr>
<tr>
<td>94</td>
<td>814</td>
<td>162</td>
<td></td>
</tr>
<tr>
<td>95</td>
<td>876</td>
<td>162</td>
<td></td>
</tr>
<tr>
<td>96</td>
<td>938</td>
<td>162</td>
<td></td>
</tr>
<tr>
<td>97</td>
<td>1000</td>
<td>162</td>
<td></td>
</tr>
<tr>
<td>98</td>
<td>1062</td>
<td>162</td>
<td></td>
</tr>
<tr>
<td>99</td>
<td>1124</td>
<td>162</td>
<td></td>
</tr>
<tr>
<td>-1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>100</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>100</td>
<td>1124</td>
<td>154</td>
<td></td>
</tr>
<tr>
<td>101</td>
<td>1062</td>
<td>154</td>
<td></td>
</tr>
<tr>
<td>102</td>
<td>1000</td>
<td>154</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>----</td>
<td>-----</td>
<td>----</td>
<td></td>
</tr>
<tr>
<td>104</td>
<td>876</td>
<td>154</td>
<td></td>
</tr>
<tr>
<td>105</td>
<td>814</td>
<td>154</td>
<td></td>
</tr>
<tr>
<td>106</td>
<td>752</td>
<td>154</td>
<td></td>
</tr>
<tr>
<td>107</td>
<td>690</td>
<td>154</td>
<td></td>
</tr>
<tr>
<td>108</td>
<td>628</td>
<td>154</td>
<td></td>
</tr>
<tr>
<td>109</td>
<td>566</td>
<td>154</td>
<td></td>
</tr>
<tr>
<td>110</td>
<td>504</td>
<td>154</td>
<td></td>
</tr>
<tr>
<td>111</td>
<td>442</td>
<td>154</td>
<td></td>
</tr>
<tr>
<td>112</td>
<td>380</td>
<td>154</td>
<td></td>
</tr>
<tr>
<td>113</td>
<td>318</td>
<td>154</td>
<td></td>
</tr>
<tr>
<td>114</td>
<td>256</td>
<td>154</td>
<td></td>
</tr>
<tr>
<td>115</td>
<td>194</td>
<td>154</td>
<td></td>
</tr>
<tr>
<td>116</td>
<td>132</td>
<td>154</td>
<td></td>
</tr>
<tr>
<td>117</td>
<td>70</td>
<td>154</td>
<td></td>
</tr>
<tr>
<td>118</td>
<td>8</td>
<td>154</td>
<td></td>
</tr>
<tr>
<td>-1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>501</td>
<td>8</td>
<td>42</td>
<td></td>
</tr>
<tr>
<td>502</td>
<td>70</td>
<td>42</td>
<td></td>
</tr>
<tr>
<td>503</td>
<td>132</td>
<td>42</td>
<td></td>
</tr>
<tr>
<td>504</td>
<td>194</td>
<td>42</td>
<td></td>
</tr>
<tr>
<td>505</td>
<td>256</td>
<td>42</td>
<td></td>
</tr>
<tr>
<td>506</td>
<td>318</td>
<td>42</td>
<td></td>
</tr>
<tr>
<td>507</td>
<td>380</td>
<td>42</td>
<td></td>
</tr>
<tr>
<td>508</td>
<td>442</td>
<td>42</td>
<td></td>
</tr>
<tr>
<td>509</td>
<td>504</td>
<td>42</td>
<td></td>
</tr>
<tr>
<td>510</td>
<td>566</td>
<td>42</td>
<td></td>
</tr>
<tr>
<td>511</td>
<td>628</td>
<td>42</td>
<td></td>
</tr>
<tr>
<td>512</td>
<td>690</td>
<td>42</td>
<td></td>
</tr>
<tr>
<td>513</td>
<td>752</td>
<td>42</td>
<td></td>
</tr>
<tr>
<td>514</td>
<td>814</td>
<td>42</td>
<td></td>
</tr>
<tr>
<td>515</td>
<td>876</td>
<td>42</td>
<td></td>
</tr>
<tr>
<td>516</td>
<td>938</td>
<td>42</td>
<td></td>
</tr>
<tr>
<td>517</td>
<td>1000</td>
<td>42</td>
<td></td>
</tr>
<tr>
<td>518</td>
<td>1062</td>
<td>42</td>
<td></td>
</tr>
<tr>
<td>519</td>
<td>1124</td>
<td>42</td>
<td></td>
</tr>
<tr>
<td>-1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>520</td>
<td>1124</td>
<td>34</td>
<td></td>
</tr>
<tr>
<td>521</td>
<td>1062</td>
<td>34</td>
<td></td>
</tr>
<tr>
<td>522</td>
<td>1000</td>
<td>34</td>
<td></td>
</tr>
<tr>
<td>523</td>
<td>938</td>
<td>34</td>
<td></td>
</tr>
<tr>
<td>524</td>
<td>876</td>
<td>34</td>
<td></td>
</tr>
<tr>
<td>525</td>
<td>814</td>
<td>34</td>
<td></td>
</tr>
<tr>
<td>526</td>
<td>752</td>
<td>34</td>
<td></td>
</tr>
<tr>
<td>527</td>
<td>690</td>
<td>34</td>
<td></td>
</tr>
<tr>
<td>528</td>
<td>628</td>
<td>34</td>
<td></td>
</tr>
<tr>
<td>529</td>
<td>566</td>
<td>34</td>
<td></td>
</tr>
<tr>
<td>530</td>
<td>504</td>
<td>34</td>
<td></td>
</tr>
<tr>
<td>531</td>
<td>442</td>
<td>34</td>
<td></td>
</tr>
<tr>
<td>532</td>
<td>380</td>
<td>34</td>
<td></td>
</tr>
<tr>
<td>533</td>
<td>318</td>
<td>34</td>
<td></td>
</tr>
<tr>
<td>534</td>
<td>256</td>
<td>34</td>
<td></td>
</tr>
<tr>
<td>535</td>
<td>194</td>
<td>34</td>
<td></td>
</tr>
<tr>
<td>536</td>
<td>132</td>
<td>34</td>
<td></td>
</tr>
<tr>
<td>537</td>
<td>70</td>
<td>34</td>
<td></td>
</tr>
<tr>
<td>538</td>
<td>8</td>
<td>34</td>
<td></td>
</tr>
<tr>
<td>-1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>541</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td></td>
</tr>
<tr>
<td>541</td>
<td>8</td>
<td>26</td>
<td></td>
</tr>
<tr>
<td>542</td>
<td>70</td>
<td>26</td>
<td></td>
</tr>
<tr>
<td>543</td>
<td>132</td>
<td>26</td>
<td></td>
</tr>
<tr>
<td>544</td>
<td>194</td>
<td>26</td>
<td></td>
</tr>
<tr>
<td>545</td>
<td>256</td>
<td>26</td>
<td></td>
</tr>
<tr>
<td>546</td>
<td>318</td>
<td>26</td>
<td></td>
</tr>
<tr>
<td>547</td>
<td>380</td>
<td>26</td>
<td></td>
</tr>
<tr>
<td>548</td>
<td>442</td>
<td>26</td>
<td></td>
</tr>
<tr>
<td>549</td>
<td>504</td>
<td>26</td>
<td></td>
</tr>
<tr>
<td>550</td>
<td>566</td>
<td>26</td>
<td></td>
</tr>
<tr>
<td>551</td>
<td>628</td>
<td>26</td>
<td></td>
</tr>
<tr>
<td>552</td>
<td>690</td>
<td>26</td>
<td></td>
</tr>
<tr>
<td>553</td>
<td>752</td>
<td>26</td>
<td></td>
</tr>
<tr>
<td>554</td>
<td>814</td>
<td>26</td>
<td></td>
</tr>
<tr>
<td>555</td>
<td>876</td>
<td>26</td>
<td></td>
</tr>
<tr>
<td>556</td>
<td>938</td>
<td>26</td>
<td></td>
</tr>
<tr>
<td>557</td>
<td>1000</td>
<td>26</td>
<td></td>
</tr>
<tr>
<td>558</td>
<td>1062</td>
<td>26</td>
<td></td>
</tr>
<tr>
<td>559</td>
<td>1124</td>
<td>26</td>
<td></td>
</tr>
<tr>
<td>-1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>560</td>
<td>1124</td>
<td>18</td>
<td></td>
</tr>
<tr>
<td>561</td>
<td>1062</td>
<td>18</td>
<td></td>
</tr>
<tr>
<td>562</td>
<td>1000</td>
<td>18</td>
<td></td>
</tr>
<tr>
<td>563</td>
<td>938</td>
<td>18</td>
<td></td>
</tr>
<tr>
<td>564</td>
<td>876</td>
<td>18</td>
<td></td>
</tr>
<tr>
<td>565</td>
<td>814</td>
<td>18</td>
<td></td>
</tr>
<tr>
<td>566</td>
<td>752</td>
<td>18</td>
<td></td>
</tr>
<tr>
<td>567</td>
<td>690</td>
<td>18</td>
<td></td>
</tr>
<tr>
<td>568</td>
<td>628</td>
<td>18</td>
<td></td>
</tr>
<tr>
<td>569</td>
<td>566</td>
<td>18</td>
<td></td>
</tr>
<tr>
<td>570</td>
<td>504</td>
<td>18</td>
<td></td>
</tr>
<tr>
<td>571</td>
<td>442</td>
<td>18</td>
<td></td>
</tr>
<tr>
<td>572</td>
<td>380</td>
<td>18</td>
<td></td>
</tr>
<tr>
<td>573</td>
<td>318</td>
<td>18</td>
<td></td>
</tr>
<tr>
<td>574</td>
<td>256</td>
<td>18</td>
<td></td>
</tr>
<tr>
<td>575</td>
<td>194</td>
<td>18</td>
<td></td>
</tr>
<tr>
<td>576</td>
<td>132</td>
<td>18</td>
<td></td>
</tr>
<tr>
<td>577</td>
<td>10</td>
<td>18</td>
<td></td>
</tr>
<tr>
<td>578</td>
<td>10</td>
<td>18</td>
<td></td>
</tr>
<tr>
<td>-1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>581</td>
<td>8</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>582</td>
<td>70</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>583</td>
<td>132</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>584</td>
<td>194</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>585</td>
<td>256</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>586</td>
<td>318</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>587</td>
<td>380</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>588</td>
<td>442</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>589</td>
<td>504</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>590</td>
<td>566</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>591</td>
<td>628</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>592</td>
<td>690</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>593</td>
<td>752</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>594</td>
<td>814</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>595</td>
<td>876</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>596</td>
<td>938</td>
<td>10</td>
<td></td>
</tr>
</tbody>
</table>

@in the final version of the program 'eindversie' this maintrack is @
@not used@
<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>598</td>
<td>1062</td>
<td>10</td>
</tr>
<tr>
<td>599</td>
<td>1124</td>
<td>10</td>
</tr>
<tr>
<td>-1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>600</td>
<td></td>
<td></td>
</tr>
<tr>
<td>601</td>
<td></td>
<td></td>
</tr>
<tr>
<td>602</td>
<td></td>
<td></td>
</tr>
<tr>
<td>603</td>
<td></td>
<td></td>
</tr>
<tr>
<td>604</td>
<td></td>
<td></td>
</tr>
<tr>
<td>605</td>
<td></td>
<td></td>
</tr>
<tr>
<td>606</td>
<td></td>
<td></td>
</tr>
<tr>
<td>607</td>
<td></td>
<td></td>
</tr>
<tr>
<td>608</td>
<td></td>
<td></td>
</tr>
<tr>
<td>609</td>
<td></td>
<td></td>
</tr>
<tr>
<td>610</td>
<td></td>
<td></td>
</tr>
<tr>
<td>611</td>
<td></td>
<td></td>
</tr>
<tr>
<td>612</td>
<td></td>
<td></td>
</tr>
<tr>
<td>613</td>
<td></td>
<td></td>
</tr>
<tr>
<td>614</td>
<td></td>
<td></td>
</tr>
<tr>
<td>615</td>
<td></td>
<td></td>
</tr>
<tr>
<td>616</td>
<td></td>
<td></td>
</tr>
<tr>
<td>617</td>
<td></td>
<td></td>
</tr>
<tr>
<td>618</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

-1

@These sets of 13 represent main tracks, the stack lanes @
@nodenumber, x co-ordinate and y co-ordinate @

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td></td>
<td></td>
</tr>
<tr>
<td>202</td>
<td></td>
<td></td>
</tr>
<tr>
<td>203</td>
<td></td>
<td></td>
</tr>
<tr>
<td>204</td>
<td></td>
<td></td>
</tr>
<tr>
<td>205</td>
<td></td>
<td></td>
</tr>
<tr>
<td>206</td>
<td></td>
<td></td>
</tr>
<tr>
<td>207</td>
<td></td>
<td></td>
</tr>
<tr>
<td>208</td>
<td></td>
<td></td>
</tr>
<tr>
<td>209</td>
<td></td>
<td></td>
</tr>
<tr>
<td>210</td>
<td></td>
<td></td>
</tr>
<tr>
<td>211</td>
<td></td>
<td></td>
</tr>
<tr>
<td>212</td>
<td></td>
<td></td>
</tr>
<tr>
<td>213</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

-1

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>218</td>
<td></td>
<td></td>
</tr>
<tr>
<td>219</td>
<td></td>
<td></td>
</tr>
<tr>
<td>220</td>
<td></td>
<td></td>
</tr>
<tr>
<td>221</td>
<td></td>
<td></td>
</tr>
<tr>
<td>222</td>
<td></td>
<td></td>
</tr>
<tr>
<td>223</td>
<td></td>
<td></td>
</tr>
<tr>
<td>224</td>
<td></td>
<td></td>
</tr>
<tr>
<td>225</td>
<td></td>
<td></td>
</tr>
<tr>
<td>226</td>
<td></td>
<td></td>
</tr>
<tr>
<td>227</td>
<td></td>
<td></td>
</tr>
<tr>
<td>228</td>
<td></td>
<td></td>
</tr>
<tr>
<td>229</td>
<td></td>
<td></td>
</tr>
<tr>
<td>230</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

-1

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>231</td>
<td></td>
<td></td>
</tr>
<tr>
<td>232</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>231</td>
<td>132</td>
<td>146</td>
</tr>
<tr>
<td>232</td>
<td>132</td>
<td>134</td>
</tr>
<tr>
<td>233</td>
<td>132</td>
<td>131</td>
</tr>
<tr>
<td>234</td>
<td>132</td>
<td>119</td>
</tr>
<tr>
<td>235</td>
<td>132</td>
<td>116</td>
</tr>
<tr>
<td>236</td>
<td>132</td>
<td>104</td>
</tr>
<tr>
<td>237</td>
<td>132</td>
<td>101</td>
</tr>
<tr>
<td>238</td>
<td>132</td>
<td>89</td>
</tr>
<tr>
<td>239</td>
<td>132</td>
<td>86</td>
</tr>
<tr>
<td>240</td>
<td>132</td>
<td>74</td>
</tr>
<tr>
<td>241</td>
<td>132</td>
<td>66</td>
</tr>
<tr>
<td>242</td>
<td>132</td>
<td>58</td>
</tr>
<tr>
<td>243</td>
<td>132</td>
<td>50</td>
</tr>
<tr>
<td>-1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>248</td>
<td></td>
<td></td>
</tr>
<tr>
<td>249</td>
<td>194</td>
<td>50</td>
</tr>
<tr>
<td>250</td>
<td>194</td>
<td>58</td>
</tr>
<tr>
<td>251</td>
<td>194</td>
<td>66</td>
</tr>
<tr>
<td>252</td>
<td>194</td>
<td>74</td>
</tr>
<tr>
<td>253</td>
<td>194</td>
<td>86</td>
</tr>
<tr>
<td>254</td>
<td>194</td>
<td>89</td>
</tr>
<tr>
<td>255</td>
<td>194</td>
<td>101</td>
</tr>
<tr>
<td>256</td>
<td>194</td>
<td>104</td>
</tr>
<tr>
<td>257</td>
<td>194</td>
<td>116</td>
</tr>
<tr>
<td>258</td>
<td>194</td>
<td>119</td>
</tr>
<tr>
<td>259</td>
<td>194</td>
<td>131</td>
</tr>
<tr>
<td>260</td>
<td>194</td>
<td>134</td>
</tr>
<tr>
<td>-1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>261</td>
<td></td>
<td></td>
</tr>
<tr>
<td>262</td>
<td>256</td>
<td>146</td>
</tr>
<tr>
<td>263</td>
<td>256</td>
<td>134</td>
</tr>
<tr>
<td>264</td>
<td>256</td>
<td>131</td>
</tr>
<tr>
<td>265</td>
<td>256</td>
<td>119</td>
</tr>
<tr>
<td>266</td>
<td>256</td>
<td>116</td>
</tr>
<tr>
<td>267</td>
<td>256</td>
<td>104</td>
</tr>
<tr>
<td>268</td>
<td>256</td>
<td>101</td>
</tr>
<tr>
<td>269</td>
<td>256</td>
<td>89</td>
</tr>
<tr>
<td>270</td>
<td>256</td>
<td>86</td>
</tr>
<tr>
<td>271</td>
<td>256</td>
<td>74</td>
</tr>
<tr>
<td>272</td>
<td>256</td>
<td>66</td>
</tr>
<tr>
<td>273</td>
<td>256</td>
<td>58</td>
</tr>
<tr>
<td>-1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>278</td>
<td></td>
<td></td>
</tr>
<tr>
<td>279</td>
<td>318</td>
<td>50</td>
</tr>
<tr>
<td>280</td>
<td>318</td>
<td>58</td>
</tr>
<tr>
<td>281</td>
<td>318</td>
<td>66</td>
</tr>
<tr>
<td>282</td>
<td>318</td>
<td>74</td>
</tr>
<tr>
<td>283</td>
<td>318</td>
<td>86</td>
</tr>
<tr>
<td>284</td>
<td>318</td>
<td>89</td>
</tr>
<tr>
<td>285</td>
<td>318</td>
<td>101</td>
</tr>
<tr>
<td>286</td>
<td>318</td>
<td>104</td>
</tr>
<tr>
<td>287</td>
<td>318</td>
<td>116</td>
</tr>
<tr>
<td>288</td>
<td>318</td>
<td>119</td>
</tr>
<tr>
<td>289</td>
<td>318</td>
<td>131</td>
</tr>
<tr>
<td>290</td>
<td>318</td>
<td>134</td>
</tr>
<tr>
<td>-1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>291</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
412  876  134
413  876  131
414  876  119
415  876  116
416  876  104
417  876  101
418  876  89
419  876  86
420  876  74
421  876  66
422  876  58
-1
428  938  50
429  938  58
430  938  66
431  938  74
432  938  86
433  938  89
434  938  101
435  938  104
436  938  116
437  938  119
438  938  131
439  938  134
440  938  146
-1
441  1000  146
442  1000  134
443  1000  131
444  1000  119
445  1000  116
446  1000  104
447  1000  101
448  1000  89
449  1000  86
450  1000  74
451  1000  66
452  1000  58
453  1000  50
-1
458  1062  50 @in the final version of the program 'eindversie' this stacklane is @
459  1062  58 @not used@
460  1062  66
461  1062  74
462  1062  86
463  1062  89
464  1062  101
465  1062  104
466  1062  116
467  1062  119
468  1062  131
469  1062  134
470  1062  146
-1
471  1124  146 @in the final version of the program 'eindversie' this stacklane is @
<table>
<thead>
<tr>
<th>Stack</th>
<th>Node In</th>
<th>Node Out</th>
<th>X Coordinate</th>
<th>Y Coordinate</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stack1</td>
<td>201</td>
<td>230</td>
<td>39</td>
<td>146</td>
<td>In the final version of the program 'eindversie' this stack lane is @used@.</td>
</tr>
<tr>
<td>Stack2</td>
<td>229</td>
<td>202</td>
<td>39</td>
<td>134</td>
<td>'eindversie' these stacks are not used @.</td>
</tr>
<tr>
<td>Stack3</td>
<td>203</td>
<td>228</td>
<td>39</td>
<td>131</td>
<td></td>
</tr>
<tr>
<td>Stack4</td>
<td>227</td>
<td>204</td>
<td>39</td>
<td>119</td>
<td></td>
</tr>
<tr>
<td>Stack5</td>
<td>205</td>
<td>226</td>
<td>39</td>
<td>116</td>
<td></td>
</tr>
<tr>
<td>Stack6</td>
<td>225</td>
<td>206</td>
<td>39</td>
<td>104</td>
<td></td>
</tr>
<tr>
<td>Stack7</td>
<td>207</td>
<td>224</td>
<td>39</td>
<td>101</td>
<td></td>
</tr>
<tr>
<td>Stack8</td>
<td>223</td>
<td>208</td>
<td>39</td>
<td>89</td>
<td></td>
</tr>
<tr>
<td>Stack9</td>
<td>209</td>
<td>222</td>
<td>39</td>
<td>86</td>
<td></td>
</tr>
<tr>
<td>Stack10</td>
<td>221</td>
<td>210</td>
<td>39</td>
<td>74</td>
<td></td>
</tr>
<tr>
<td>Stack11</td>
<td>230</td>
<td>231</td>
<td>101</td>
<td>146</td>
<td>In the final version of the program @.</td>
</tr>
<tr>
<td>Stack12</td>
<td>232</td>
<td>229</td>
<td>101</td>
<td>134</td>
<td>'eindversie' these stacks are not used @.</td>
</tr>
<tr>
<td>Stack13</td>
<td>228</td>
<td>233</td>
<td>101</td>
<td>131</td>
<td></td>
</tr>
<tr>
<td>Stack14</td>
<td>234</td>
<td>227</td>
<td>101</td>
<td>119</td>
<td></td>
</tr>
<tr>
<td>Stack15</td>
<td>226</td>
<td>235</td>
<td>101</td>
<td>116</td>
<td></td>
</tr>
<tr>
<td>Stack16</td>
<td>236</td>
<td>225</td>
<td>101</td>
<td>104</td>
<td></td>
</tr>
<tr>
<td>Stack17</td>
<td>224</td>
<td>237</td>
<td>101</td>
<td>101</td>
<td></td>
</tr>
<tr>
<td>Stack18</td>
<td>238</td>
<td>223</td>
<td>101</td>
<td>89</td>
<td></td>
</tr>
<tr>
<td>Stack19</td>
<td>222</td>
<td>239</td>
<td>101</td>
<td>86</td>
<td></td>
</tr>
<tr>
<td>Stack20</td>
<td>240</td>
<td>221</td>
<td>101</td>
<td>74</td>
<td></td>
</tr>
<tr>
<td>Stack21</td>
<td>231</td>
<td>260</td>
<td>163</td>
<td>146</td>
<td></td>
</tr>
<tr>
<td>Stack22</td>
<td>259</td>
<td>232</td>
<td>163</td>
<td>134</td>
<td></td>
</tr>
<tr>
<td>Stack23</td>
<td>233</td>
<td>258</td>
<td>163</td>
<td>131</td>
<td></td>
</tr>
<tr>
<td>Stack24</td>
<td>257</td>
<td>234</td>
<td>163</td>
<td>119</td>
<td></td>
</tr>
<tr>
<td>Stack25</td>
<td>235</td>
<td>256</td>
<td>163</td>
<td>116</td>
<td></td>
</tr>
<tr>
<td>Stack26</td>
<td>255</td>
<td>236</td>
<td>163</td>
<td>104</td>
<td></td>
</tr>
<tr>
<td>Stack27</td>
<td>237</td>
<td>254</td>
<td>163</td>
<td>101</td>
<td></td>
</tr>
<tr>
<td>Stack28</td>
<td>253</td>
<td>238</td>
<td>163</td>
<td>89</td>
<td></td>
</tr>
<tr>
<td>Stack29</td>
<td>239</td>
<td>252</td>
<td>163</td>
<td>86</td>
<td></td>
</tr>
<tr>
<td>Stack30</td>
<td>251</td>
<td>240</td>
<td>163</td>
<td>74</td>
<td></td>
</tr>
<tr>
<td>Stack31</td>
<td>260</td>
<td>261</td>
<td>225</td>
<td>146</td>
<td></td>
</tr>
<tr>
<td>Stack32</td>
<td>262</td>
<td>259</td>
<td>225</td>
<td>134</td>
<td></td>
</tr>
<tr>
<td>Stack33</td>
<td>258</td>
<td>263</td>
<td>225</td>
<td>131</td>
<td></td>
</tr>
<tr>
<td>Stack34</td>
<td>264</td>
<td>257</td>
<td>225</td>
<td>119</td>
<td></td>
</tr>
<tr>
<td>Stack35</td>
<td>256</td>
<td>285</td>
<td>225</td>
<td>116</td>
<td></td>
</tr>
<tr>
<td>Stack36</td>
<td>266</td>
<td>255</td>
<td>225</td>
<td>104</td>
<td></td>
</tr>
<tr>
<td>Stack37</td>
<td>254</td>
<td>267</td>
<td>225</td>
<td>101</td>
<td></td>
</tr>
<tr>
<td>Stack38</td>
<td>268</td>
<td>253</td>
<td>225</td>
<td>89</td>
<td></td>
</tr>
<tr>
<td>Stack39</td>
<td>252</td>
<td>269</td>
<td>225</td>
<td>86</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>261</td>
<td>290</td>
<td>287</td>
<td>146</td>
<td></td>
</tr>
<tr>
<td>-------</td>
<td>------</td>
<td>------</td>
<td>------</td>
<td>------</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>289</td>
<td>262</td>
<td>287</td>
<td>134</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>283</td>
<td>288</td>
<td>287</td>
<td>131</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>287</td>
<td>264</td>
<td>287</td>
<td>119</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>285</td>
<td>286</td>
<td>287</td>
<td>116</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>285</td>
<td>286</td>
<td>287</td>
<td>104</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>267</td>
<td>284</td>
<td>287</td>
<td>101</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>283</td>
<td>288</td>
<td>287</td>
<td>89</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>289</td>
<td>282</td>
<td>287</td>
<td>86</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>281</td>
<td>270</td>
<td>287</td>
<td>74</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>290</td>
<td>291</td>
<td>349</td>
<td>146</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>292</td>
<td>289</td>
<td>349</td>
<td>134</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>288</td>
<td>293</td>
<td>349</td>
<td>131</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>294</td>
<td>287</td>
<td>349</td>
<td>119</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>286</td>
<td>295</td>
<td>349</td>
<td>116</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>296</td>
<td>285</td>
<td>349</td>
<td>104</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>284</td>
<td>297</td>
<td>349</td>
<td>101</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>298</td>
<td>283</td>
<td>349</td>
<td>89</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>282</td>
<td>299</td>
<td>349</td>
<td>86</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>300</td>
<td>281</td>
<td>349</td>
<td>74</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>291</td>
<td>320</td>
<td>411</td>
<td>146</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>319</td>
<td>292</td>
<td>411</td>
<td>134</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>293</td>
<td>318</td>
<td>411</td>
<td>131</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>317</td>
<td>294</td>
<td>411</td>
<td>119</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>295</td>
<td>316</td>
<td>411</td>
<td>116</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>315</td>
<td>296</td>
<td>411</td>
<td>104</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>297</td>
<td>314</td>
<td>411</td>
<td>101</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>313</td>
<td>298</td>
<td>411</td>
<td>89</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>299</td>
<td>312</td>
<td>411</td>
<td>86</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>311</td>
<td>300</td>
<td>411</td>
<td>74</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>320</td>
<td>321</td>
<td>473</td>
<td>146</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>322</td>
<td>319</td>
<td>473</td>
<td>134</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>318</td>
<td>323</td>
<td>473</td>
<td>131</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>324</td>
<td>317</td>
<td>473</td>
<td>119</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>316</td>
<td>325</td>
<td>473</td>
<td>116</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>326</td>
<td>315</td>
<td>473</td>
<td>104</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>314</td>
<td>327</td>
<td>473</td>
<td>101</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>328</td>
<td>313</td>
<td>473</td>
<td>89</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>312</td>
<td>329</td>
<td>473</td>
<td>86</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>330</td>
<td>311</td>
<td>473</td>
<td>74</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>321</td>
<td>350</td>
<td>535</td>
<td>146</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>349</td>
<td>322</td>
<td>535</td>
<td>134</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>323</td>
<td>348</td>
<td>535</td>
<td>131</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>347</td>
<td>324</td>
<td>535</td>
<td>119</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>325</td>
<td>346</td>
<td>535</td>
<td>116</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>345</td>
<td>326</td>
<td>535</td>
<td>104</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>327</td>
<td>344</td>
<td>535</td>
<td>101</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>343</td>
<td>328</td>
<td>535</td>
<td>89</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>329</td>
<td>342</td>
<td>535</td>
<td>86</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>341</td>
<td>330</td>
<td>535</td>
<td>74</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>350</td>
<td>351</td>
<td>597</td>
<td>146</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>352</td>
<td>349</td>
<td>597</td>
<td>134</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>348</td>
<td>353</td>
<td>597</td>
<td>131</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>354</td>
<td>347</td>
<td>597</td>
<td>119</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>354</td>
<td>347</td>
<td>597</td>
<td>119</td>
<td></td>
</tr>
<tr>
<td>-------</td>
<td>-----</td>
<td>-----</td>
<td>-----</td>
<td>-----</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>346</td>
<td>355</td>
<td>597</td>
<td>116</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>356</td>
<td>345</td>
<td>597</td>
<td>104</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>344</td>
<td>357</td>
<td>597</td>
<td>101</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>358</td>
<td>343</td>
<td>597</td>
<td>89</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>342</td>
<td>359</td>
<td>597</td>
<td>86</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>360</td>
<td>341</td>
<td>597</td>
<td>74</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>351</td>
<td>380</td>
<td>659</td>
<td>146</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>379</td>
<td>352</td>
<td>659</td>
<td>134</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>353</td>
<td>378</td>
<td>659</td>
<td>131</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>377</td>
<td>354</td>
<td>659</td>
<td>119</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>355</td>
<td>376</td>
<td>659</td>
<td>116</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>375</td>
<td>356</td>
<td>659</td>
<td>104</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>357</td>
<td>374</td>
<td>659</td>
<td>101</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>373</td>
<td>358</td>
<td>659</td>
<td>89</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>359</td>
<td>372</td>
<td>659</td>
<td>86</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>371</td>
<td>360</td>
<td>659</td>
<td>74</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>380</td>
<td>381</td>
<td>721</td>
<td>146</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>382</td>
<td>379</td>
<td>721</td>
<td>134</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>378</td>
<td>383</td>
<td>721</td>
<td>131</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>384</td>
<td>377</td>
<td>721</td>
<td>119</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>376</td>
<td>385</td>
<td>721</td>
<td>116</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>386</td>
<td>375</td>
<td>721</td>
<td>104</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>374</td>
<td>387</td>
<td>721</td>
<td>101</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>388</td>
<td>373</td>
<td>721</td>
<td>89</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>372</td>
<td>389</td>
<td>721</td>
<td>86</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>390</td>
<td>371</td>
<td>721</td>
<td>74</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>381</td>
<td>410</td>
<td>783</td>
<td>146</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>409</td>
<td>382</td>
<td>783</td>
<td>134</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>383</td>
<td>408</td>
<td>783</td>
<td>131</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>407</td>
<td>384</td>
<td>783</td>
<td>119</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>385</td>
<td>406</td>
<td>783</td>
<td>116</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>405</td>
<td>386</td>
<td>783</td>
<td>104</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>387</td>
<td>404</td>
<td>783</td>
<td>101</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>403</td>
<td>388</td>
<td>783</td>
<td>89</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>389</td>
<td>402</td>
<td>783</td>
<td>86</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>401</td>
<td>390</td>
<td>783</td>
<td>74</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>410</td>
<td>411</td>
<td>845</td>
<td>146</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>412</td>
<td>409</td>
<td>845</td>
<td>134</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>408</td>
<td>413</td>
<td>845</td>
<td>131</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>414</td>
<td>407</td>
<td>845</td>
<td>119</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>406</td>
<td>415</td>
<td>845</td>
<td>116</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>416</td>
<td>405</td>
<td>845</td>
<td>104</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>404</td>
<td>417</td>
<td>845</td>
<td>101</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>418</td>
<td>403</td>
<td>845</td>
<td>89</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>402</td>
<td>419</td>
<td>845</td>
<td>86</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>420</td>
<td>401</td>
<td>845</td>
<td>74</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>411</td>
<td>440</td>
<td>907</td>
<td>146</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>439</td>
<td>412</td>
<td>907</td>
<td>134</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>413</td>
<td>438</td>
<td>907</td>
<td>131</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>437</td>
<td>414</td>
<td>907</td>
<td>119</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>415</td>
<td>436</td>
<td>907</td>
<td>116</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>435</td>
<td>416</td>
<td>907</td>
<td>104</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>417</td>
<td>434</td>
<td>907</td>
<td>101</td>
<td></td>
</tr>
<tr>
<td>Stack</td>
<td>433</td>
<td>418</td>
<td>907</td>
<td>89</td>
<td></td>
</tr>
<tr>
<td>stack</td>
<td>431</td>
<td>420</td>
<td>907</td>
<td>74</td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td>-----</td>
<td>-----</td>
<td>-----</td>
<td>----</td>
<td></td>
</tr>
<tr>
<td>stack150</td>
<td>440</td>
<td>441</td>
<td>969</td>
<td>146</td>
<td></td>
</tr>
<tr>
<td>stack151</td>
<td>442</td>
<td>439</td>
<td>969</td>
<td>134</td>
<td></td>
</tr>
<tr>
<td>stack152</td>
<td>438</td>
<td>443</td>
<td>969</td>
<td>131</td>
<td></td>
</tr>
<tr>
<td>stack153</td>
<td>444</td>
<td>437</td>
<td>969</td>
<td>119</td>
<td></td>
</tr>
<tr>
<td>stack154</td>
<td>436</td>
<td>445</td>
<td>969</td>
<td>116</td>
<td></td>
</tr>
<tr>
<td>stack155</td>
<td>446</td>
<td>435</td>
<td>969</td>
<td>104</td>
<td></td>
</tr>
<tr>
<td>stack156</td>
<td>434</td>
<td>447</td>
<td>969</td>
<td>101</td>
<td></td>
</tr>
<tr>
<td>stack157</td>
<td>448</td>
<td>433</td>
<td>969</td>
<td>89</td>
<td></td>
</tr>
<tr>
<td>stack158</td>
<td>432</td>
<td>449</td>
<td>969</td>
<td>86</td>
<td></td>
</tr>
<tr>
<td>stack159</td>
<td>450</td>
<td>431</td>
<td>969</td>
<td>74</td>
<td></td>
</tr>
<tr>
<td>stack160</td>
<td>441</td>
<td>470</td>
<td>1031</td>
<td>146</td>
<td></td>
</tr>
<tr>
<td>stack161</td>
<td>469</td>
<td>442</td>
<td>1031</td>
<td>134</td>
<td></td>
</tr>
<tr>
<td>stack162</td>
<td>443</td>
<td>468</td>
<td>1031</td>
<td>131</td>
<td></td>
</tr>
<tr>
<td>stack163</td>
<td>467</td>
<td>444</td>
<td>1031</td>
<td>119</td>
<td></td>
</tr>
<tr>
<td>stack164</td>
<td>445</td>
<td>466</td>
<td>1031</td>
<td>116</td>
<td></td>
</tr>
<tr>
<td>stack165</td>
<td>465</td>
<td>446</td>
<td>1031</td>
<td>104</td>
<td></td>
</tr>
<tr>
<td>stack166</td>
<td>447</td>
<td>464</td>
<td>1031</td>
<td>101</td>
<td></td>
</tr>
<tr>
<td>stack167</td>
<td>463</td>
<td>448</td>
<td>1031</td>
<td>89</td>
<td></td>
</tr>
<tr>
<td>stack168</td>
<td>449</td>
<td>462</td>
<td>1031</td>
<td>86</td>
<td></td>
</tr>
<tr>
<td>stack169</td>
<td>461</td>
<td>450</td>
<td>1031</td>
<td>74</td>
<td></td>
</tr>
<tr>
<td>stack170</td>
<td>470</td>
<td>471</td>
<td>1093</td>
<td>146</td>
<td></td>
</tr>
<tr>
<td>stack171</td>
<td>472</td>
<td>469</td>
<td>1093</td>
<td>134</td>
<td></td>
</tr>
<tr>
<td>stack172</td>
<td>468</td>
<td>473</td>
<td>1093</td>
<td>131</td>
<td></td>
</tr>
<tr>
<td>stack173</td>
<td>474</td>
<td>467</td>
<td>1093</td>
<td>119</td>
<td></td>
</tr>
<tr>
<td>stack174</td>
<td>466</td>
<td>475</td>
<td>1093</td>
<td>116</td>
<td></td>
</tr>
<tr>
<td>stack175</td>
<td>476</td>
<td>465</td>
<td>1093</td>
<td>104</td>
<td></td>
</tr>
<tr>
<td>stack176</td>
<td>464</td>
<td>477</td>
<td>1093</td>
<td>101</td>
<td></td>
</tr>
<tr>
<td>stack177</td>
<td>478</td>
<td>463</td>
<td>1093</td>
<td>89</td>
<td></td>
</tr>
<tr>
<td>stack178</td>
<td>462</td>
<td>479</td>
<td>1093</td>
<td>86</td>
<td></td>
</tr>
<tr>
<td>stack179</td>
<td>480</td>
<td>461</td>
<td>1093</td>
<td>74</td>
<td></td>
</tr>
</tbody>
</table>

END

@portainer@  
@name, node_in, node_out, x co-ordinate and y co-ordinate@

<table>
<thead>
<tr>
<th>loc1los</th>
<th>212</th>
<th>39</th>
<th>58</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>loc1laad</td>
<td>218</td>
<td>213</td>
<td>39</td>
<td>50</td>
</tr>
<tr>
<td>loc2los</td>
<td>242</td>
<td>101</td>
<td>58</td>
<td></td>
</tr>
<tr>
<td>@loc2laad</td>
<td>218</td>
<td>243</td>
<td>101</td>
<td>50</td>
</tr>
<tr>
<td>loc3los</td>
<td>242</td>
<td>163</td>
<td>58</td>
<td></td>
</tr>
<tr>
<td>loc3laad</td>
<td>248</td>
<td>243</td>
<td>163</td>
<td>50</td>
</tr>
<tr>
<td>loc4los</td>
<td>272</td>
<td>225</td>
<td>58</td>
<td></td>
</tr>
<tr>
<td>@loc4laad</td>
<td>248</td>
<td>273</td>
<td>225</td>
<td>50</td>
</tr>
<tr>
<td>loc5los</td>
<td>272</td>
<td>287</td>
<td>58</td>
<td></td>
</tr>
<tr>
<td>loc5laad</td>
<td>278</td>
<td>273</td>
<td>287</td>
<td>50</td>
</tr>
<tr>
<td>loc6los</td>
<td>302</td>
<td>349</td>
<td>58</td>
<td></td>
</tr>
<tr>
<td>@loc6laad</td>
<td>278</td>
<td>303</td>
<td>349</td>
<td>50</td>
</tr>
<tr>
<td>loc7los</td>
<td>302</td>
<td>411</td>
<td>58</td>
<td></td>
</tr>
</tbody>
</table>
loc7los 302 309 411 58
loc7laad 308 303 411 50
loc8los 332 309 473 58
@loc8laad 308 333 473 50@
loc9los 332 339 535 58
loc9laad 338 333 535 50
loc10los 362 339 597 58
@loc10laad 338 363 597 50@
loc11los 362 369 659 58
loc11laad 368 363 659 50
loc12los 392 369 721 58
@loc12laad 368 393 721 50@
loc13los 392 399 783 58
loc13laad 398 393 783 50
loc14los 422 399 845 58
@loc14laad 398 423 845 50@
loc15los 422 429 907 58
loc15laad 428 423 907 50
loc16los 452 429 969 58
@loc16laad 428 453 969 50@
loc17los 452 459 1031 58
loc17laad 458 453 1031 50
loc18los 482 459 1093 58
@loc18laad 458 483 1093 50@

END

@short cuts@
@entrance_node of the short cut, exit_node of the short cut@
211 220
241 250
271 280
301 310
331 340
361 370
391 400
421 430
451 460
38 41
<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>78</td>
<td>81</td>
<td></td>
</tr>
<tr>
<td>81</td>
<td>118</td>
<td></td>
</tr>
<tr>
<td>118</td>
<td>201</td>
<td></td>
</tr>
<tr>
<td>230</td>
<td>117</td>
<td></td>
</tr>
<tr>
<td>117</td>
<td>82</td>
<td></td>
</tr>
<tr>
<td>82</td>
<td>77</td>
<td></td>
</tr>
<tr>
<td>77</td>
<td>42</td>
<td></td>
</tr>
<tr>
<td>42</td>
<td>37</td>
<td></td>
</tr>
<tr>
<td>37</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>36</td>
<td></td>
</tr>
<tr>
<td>36</td>
<td>43</td>
<td></td>
</tr>
<tr>
<td>43</td>
<td>76</td>
<td></td>
</tr>
<tr>
<td>76</td>
<td>83</td>
<td></td>
</tr>
<tr>
<td>83</td>
<td>116</td>
<td></td>
</tr>
<tr>
<td>116</td>
<td>231</td>
<td></td>
</tr>
<tr>
<td>260</td>
<td>115</td>
<td></td>
</tr>
<tr>
<td>115</td>
<td>84</td>
<td></td>
</tr>
<tr>
<td>84</td>
<td>75</td>
<td></td>
</tr>
<tr>
<td>75</td>
<td>44</td>
<td></td>
</tr>
<tr>
<td>44</td>
<td>35</td>
<td></td>
</tr>
<tr>
<td>35</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>34</td>
<td></td>
</tr>
<tr>
<td>34</td>
<td>45</td>
<td></td>
</tr>
<tr>
<td>45</td>
<td>74</td>
<td></td>
</tr>
<tr>
<td>74</td>
<td>85</td>
<td></td>
</tr>
<tr>
<td>85</td>
<td>114</td>
<td></td>
</tr>
<tr>
<td>114</td>
<td>261</td>
<td></td>
</tr>
<tr>
<td>290</td>
<td>113</td>
<td></td>
</tr>
<tr>
<td>113</td>
<td>86</td>
<td></td>
</tr>
<tr>
<td>86</td>
<td>73</td>
<td></td>
</tr>
<tr>
<td>73</td>
<td>46</td>
<td></td>
</tr>
<tr>
<td>46</td>
<td>33</td>
<td></td>
</tr>
<tr>
<td>33</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>32</td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>47</td>
<td></td>
</tr>
<tr>
<td>47</td>
<td>72</td>
<td></td>
</tr>
<tr>
<td>72</td>
<td>87</td>
<td></td>
</tr>
<tr>
<td>87</td>
<td>112</td>
<td></td>
</tr>
<tr>
<td>112</td>
<td>291</td>
<td></td>
</tr>
<tr>
<td>320</td>
<td>111</td>
<td></td>
</tr>
<tr>
<td>111</td>
<td>88</td>
<td></td>
</tr>
<tr>
<td>88</td>
<td>71</td>
<td></td>
</tr>
<tr>
<td>71</td>
<td>48</td>
<td></td>
</tr>
<tr>
<td>48</td>
<td>31</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>49</td>
<td></td>
</tr>
<tr>
<td>49</td>
<td>70</td>
<td></td>
</tr>
<tr>
<td>70</td>
<td>89</td>
<td></td>
</tr>
<tr>
<td>89</td>
<td>110</td>
<td></td>
</tr>
<tr>
<td>110</td>
<td>321</td>
<td></td>
</tr>
<tr>
<td>611</td>
<td>588</td>
<td></td>
</tr>
<tr>
<td>-----</td>
<td>-----</td>
<td></td>
</tr>
<tr>
<td>588</td>
<td>571</td>
<td></td>
</tr>
<tr>
<td>571</td>
<td>548</td>
<td></td>
</tr>
<tr>
<td>548</td>
<td>531</td>
<td></td>
</tr>
<tr>
<td>531</td>
<td>508</td>
<td></td>
</tr>
<tr>
<td>508</td>
<td>308</td>
<td></td>
</tr>
<tr>
<td>333</td>
<td>509</td>
<td></td>
</tr>
<tr>
<td>509</td>
<td>530</td>
<td></td>
</tr>
<tr>
<td>530</td>
<td>549</td>
<td></td>
</tr>
<tr>
<td>549</td>
<td>570</td>
<td></td>
</tr>
<tr>
<td>570</td>
<td>589</td>
<td></td>
</tr>
<tr>
<td>589</td>
<td>610</td>
<td></td>
</tr>
<tr>
<td>609</td>
<td>590</td>
<td></td>
</tr>
<tr>
<td>590</td>
<td>569</td>
<td></td>
</tr>
<tr>
<td>569</td>
<td>550</td>
<td></td>
</tr>
<tr>
<td>550</td>
<td>529</td>
<td></td>
</tr>
<tr>
<td>529</td>
<td>510</td>
<td></td>
</tr>
<tr>
<td>510</td>
<td>338</td>
<td></td>
</tr>
<tr>
<td>363</td>
<td>511</td>
<td></td>
</tr>
<tr>
<td>511</td>
<td>528</td>
<td></td>
</tr>
<tr>
<td>528</td>
<td>551</td>
<td></td>
</tr>
<tr>
<td>551</td>
<td>568</td>
<td></td>
</tr>
<tr>
<td>568</td>
<td>591</td>
<td></td>
</tr>
<tr>
<td>591</td>
<td>608</td>
<td></td>
</tr>
<tr>
<td>607</td>
<td>592</td>
<td></td>
</tr>
<tr>
<td>592</td>
<td>567</td>
<td></td>
</tr>
<tr>
<td>567</td>
<td>552</td>
<td></td>
</tr>
<tr>
<td>552</td>
<td>527</td>
<td></td>
</tr>
<tr>
<td>527</td>
<td>512</td>
<td></td>
</tr>
<tr>
<td>512</td>
<td>368</td>
<td></td>
</tr>
<tr>
<td>393</td>
<td>513</td>
<td></td>
</tr>
<tr>
<td>513</td>
<td>526</td>
<td></td>
</tr>
<tr>
<td>526</td>
<td>553</td>
<td></td>
</tr>
<tr>
<td>553</td>
<td>566</td>
<td></td>
</tr>
<tr>
<td>566</td>
<td>593</td>
<td></td>
</tr>
<tr>
<td>593</td>
<td>606</td>
<td></td>
</tr>
<tr>
<td>605</td>
<td>594</td>
<td></td>
</tr>
<tr>
<td>594</td>
<td>565</td>
<td></td>
</tr>
<tr>
<td>565</td>
<td>554</td>
<td></td>
</tr>
<tr>
<td>554</td>
<td>525</td>
<td></td>
</tr>
<tr>
<td>525</td>
<td>514</td>
<td></td>
</tr>
<tr>
<td>514</td>
<td>398</td>
<td></td>
</tr>
<tr>
<td>423</td>
<td>515</td>
<td></td>
</tr>
<tr>
<td>515</td>
<td>524</td>
<td></td>
</tr>
<tr>
<td>524</td>
<td>555</td>
<td></td>
</tr>
<tr>
<td>555</td>
<td>564</td>
<td></td>
</tr>
<tr>
<td>564</td>
<td>595</td>
<td></td>
</tr>
<tr>
<td>595</td>
<td>604</td>
<td></td>
</tr>
<tr>
<td>603</td>
<td>596</td>
<td></td>
</tr>
<tr>
<td>596</td>
<td>563</td>
<td></td>
</tr>
<tr>
<td>563</td>
<td>556</td>
<td></td>
</tr>
<tr>
<td>556</td>
<td>523</td>
<td></td>
</tr>
</tbody>
</table>
516  428
453  517
517  522
522  557
557  562
562  597
597  602
601  598
598  561
561  558
558  521
521  518
518  458
483  519
519  520
520  559
559  560
560  599
599  600
481  460

-1

@super shortcuts@
242  219
272  249
302  279
332  309
362  339
392  369
422  399
452  429
482  459

241  220
271  250
301  280
331  310
361  340
391  370
421  400
451  430
481  460

230  231
260  261
290  291
320  321
350  351
380  381
410  411
440  441
470  471

218  243
248  273
278  303
<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>278</td>
<td>303</td>
</tr>
<tr>
<td>308</td>
<td>333</td>
</tr>
<tr>
<td>338</td>
<td>363</td>
</tr>
<tr>
<td>368</td>
<td>393</td>
</tr>
<tr>
<td>398</td>
<td>423</td>
</tr>
<tr>
<td>428</td>
<td>453</td>
</tr>
<tr>
<td>458</td>
<td>483</td>
</tr>
</tbody>
</table>

-1
201
201
### Appendix 28-5. Input file for the lay out of the terminal

@ These sets of 19 represent main tracks, the inter berth lanes @
@ node number, x co-ordinate and y co-ordinate @

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>8</td>
<td>194 @ in the final version of the program 'eindversie' this maintrack is @</td>
</tr>
<tr>
<td>2</td>
<td>70</td>
<td>194 @ not used. @</td>
</tr>
<tr>
<td>3</td>
<td>132</td>
<td>194</td>
</tr>
<tr>
<td>4</td>
<td>194</td>
<td>194</td>
</tr>
<tr>
<td>5</td>
<td>256</td>
<td>194</td>
</tr>
<tr>
<td>6</td>
<td>318</td>
<td>194</td>
</tr>
<tr>
<td>7</td>
<td>380</td>
<td>194</td>
</tr>
<tr>
<td>8</td>
<td>442</td>
<td>194</td>
</tr>
<tr>
<td>9</td>
<td>504</td>
<td>194</td>
</tr>
<tr>
<td>10</td>
<td>566</td>
<td>194</td>
</tr>
<tr>
<td>11</td>
<td>628</td>
<td>194</td>
</tr>
<tr>
<td>12</td>
<td>690</td>
<td>194</td>
</tr>
<tr>
<td>13</td>
<td>752</td>
<td>194</td>
</tr>
<tr>
<td>14</td>
<td>814</td>
<td>194</td>
</tr>
<tr>
<td>15</td>
<td>876</td>
<td>194</td>
</tr>
<tr>
<td>16</td>
<td>938</td>
<td>194</td>
</tr>
<tr>
<td>17</td>
<td>1000</td>
<td>194</td>
</tr>
<tr>
<td>18</td>
<td>1062</td>
<td>194</td>
</tr>
<tr>
<td>19</td>
<td>1124</td>
<td>194</td>
</tr>
<tr>
<td>-1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>1124</td>
<td>186 @ in the final version of the program 'eindversie' this maintrack is @</td>
</tr>
<tr>
<td>21</td>
<td>1062</td>
<td>186 @ not used. @</td>
</tr>
<tr>
<td>22</td>
<td>1000</td>
<td>186</td>
</tr>
<tr>
<td>23</td>
<td>938</td>
<td>186</td>
</tr>
<tr>
<td>24</td>
<td>876</td>
<td>186</td>
</tr>
<tr>
<td>25</td>
<td>814</td>
<td>186</td>
</tr>
<tr>
<td>26</td>
<td>752</td>
<td>186</td>
</tr>
<tr>
<td>27</td>
<td>690</td>
<td>186</td>
</tr>
<tr>
<td>28</td>
<td>628</td>
<td>186</td>
</tr>
<tr>
<td>29</td>
<td>566</td>
<td>186</td>
</tr>
<tr>
<td>30</td>
<td>504</td>
<td>186</td>
</tr>
<tr>
<td>31</td>
<td>442</td>
<td>186</td>
</tr>
<tr>
<td>32</td>
<td>380</td>
<td>186</td>
</tr>
<tr>
<td>33</td>
<td>318</td>
<td>186</td>
</tr>
<tr>
<td>34</td>
<td>256</td>
<td>186</td>
</tr>
<tr>
<td>35</td>
<td>194</td>
<td>186</td>
</tr>
<tr>
<td>36</td>
<td>132</td>
<td>186</td>
</tr>
<tr>
<td>37</td>
<td>70</td>
<td>186</td>
</tr>
<tr>
<td>38</td>
<td>8</td>
<td>186</td>
</tr>
<tr>
<td>-1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>41</td>
<td>8</td>
<td>178</td>
</tr>
<tr>
<td>42</td>
<td>70</td>
<td>178</td>
</tr>
<tr>
<td>43</td>
<td>132</td>
<td>178</td>
</tr>
<tr>
<td>44</td>
<td>194</td>
<td>178</td>
</tr>
<tr>
<td>45</td>
<td>256</td>
<td>178</td>
</tr>
<tr>
<td>46</td>
<td>318</td>
<td>178</td>
</tr>
<tr>
<td>47</td>
<td>380</td>
<td>178</td>
</tr>
<tr>
<td>48</td>
<td>442</td>
<td>178</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>-----</td>
<td>-----</td>
</tr>
<tr>
<td>49</td>
<td>504</td>
<td>178</td>
</tr>
<tr>
<td>50</td>
<td>566</td>
<td>178</td>
</tr>
<tr>
<td>51</td>
<td>628</td>
<td>178</td>
</tr>
<tr>
<td>52</td>
<td>690</td>
<td>178</td>
</tr>
<tr>
<td>53</td>
<td>752</td>
<td>178</td>
</tr>
<tr>
<td>54</td>
<td>814</td>
<td>178</td>
</tr>
<tr>
<td>55</td>
<td>876</td>
<td>178</td>
</tr>
<tr>
<td>56</td>
<td>938</td>
<td>178</td>
</tr>
<tr>
<td>57</td>
<td>1000</td>
<td>178</td>
</tr>
<tr>
<td>58</td>
<td>1062</td>
<td>178</td>
</tr>
<tr>
<td>59</td>
<td>1124</td>
<td>178</td>
</tr>
<tr>
<td>60</td>
<td></td>
<td></td>
</tr>
<tr>
<td>60</td>
<td>1124</td>
<td>170</td>
</tr>
<tr>
<td>61</td>
<td>1062</td>
<td>170</td>
</tr>
<tr>
<td>62</td>
<td>1000</td>
<td>170</td>
</tr>
<tr>
<td>63</td>
<td>938</td>
<td>170</td>
</tr>
<tr>
<td>64</td>
<td>876</td>
<td>170</td>
</tr>
<tr>
<td>65</td>
<td>814</td>
<td>170</td>
</tr>
<tr>
<td>66</td>
<td>752</td>
<td>170</td>
</tr>
<tr>
<td>67</td>
<td>690</td>
<td>170</td>
</tr>
<tr>
<td>68</td>
<td>628</td>
<td>170</td>
</tr>
<tr>
<td>69</td>
<td>566</td>
<td>170</td>
</tr>
<tr>
<td>70</td>
<td>504</td>
<td>170</td>
</tr>
<tr>
<td>71</td>
<td>442</td>
<td>170</td>
</tr>
<tr>
<td>72</td>
<td>380</td>
<td>170</td>
</tr>
<tr>
<td>73</td>
<td>318</td>
<td>170</td>
</tr>
<tr>
<td>74</td>
<td>256</td>
<td>170</td>
</tr>
<tr>
<td>75</td>
<td>194</td>
<td>170</td>
</tr>
<tr>
<td>76</td>
<td>132</td>
<td>170</td>
</tr>
<tr>
<td>77</td>
<td>70</td>
<td>170</td>
</tr>
<tr>
<td>78</td>
<td>8</td>
<td>170</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>81</td>
<td>8</td>
<td>162</td>
</tr>
<tr>
<td>82</td>
<td>70</td>
<td>162</td>
</tr>
<tr>
<td>83</td>
<td>132</td>
<td>162</td>
</tr>
<tr>
<td>84</td>
<td>194</td>
<td>162</td>
</tr>
<tr>
<td>85</td>
<td>256</td>
<td>162</td>
</tr>
<tr>
<td>86</td>
<td>318</td>
<td>162</td>
</tr>
<tr>
<td>87</td>
<td>380</td>
<td>162</td>
</tr>
<tr>
<td>88</td>
<td>442</td>
<td>162</td>
</tr>
<tr>
<td>89</td>
<td>504</td>
<td>162</td>
</tr>
<tr>
<td>90</td>
<td>566</td>
<td>162</td>
</tr>
<tr>
<td>91</td>
<td>628</td>
<td>162</td>
</tr>
<tr>
<td>92</td>
<td>690</td>
<td>162</td>
</tr>
<tr>
<td>93</td>
<td>752</td>
<td>162</td>
</tr>
<tr>
<td>94</td>
<td>814</td>
<td>162</td>
</tr>
<tr>
<td>95</td>
<td>876</td>
<td>162</td>
</tr>
<tr>
<td>96</td>
<td>938</td>
<td>162</td>
</tr>
<tr>
<td>97</td>
<td>1000</td>
<td>162</td>
</tr>
<tr>
<td>98</td>
<td>1062</td>
<td>162</td>
</tr>
<tr>
<td>99</td>
<td>1124</td>
<td>162</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>100</td>
<td>1124</td>
<td>154</td>
</tr>
<tr>
<td>101</td>
<td>1062</td>
<td>154</td>
</tr>
<tr>
<td>102</td>
<td>1000</td>
<td>154</td>
</tr>
</tbody>
</table>
Appendix 28-VI Input file Rgv attributes

Using an input file the attributes of the rgv can be changed. The listing of the input file is shown below. This input file is called trolley_conf.txt.

@configuration in file trolley_conf.txt@

@Rgv speed in m/s on main route @ 3
@Rgv speed in m/s passing stack lane @ 1
@Rgv speed in m/s passing short cut@ 2

@handling time in seconds to pick up or deliver a load@ 30
@minimum distance between trolleys on the main track@ 3