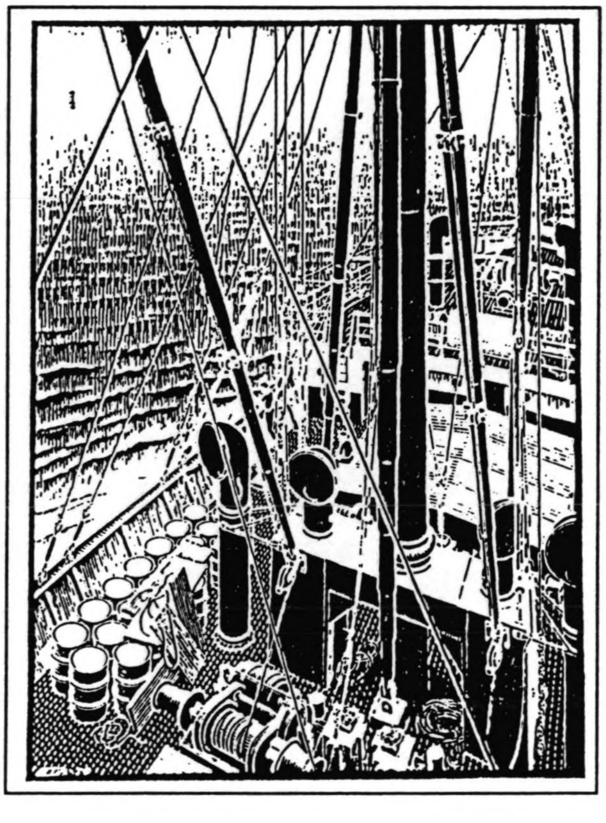
Simulation Model for Port Operations; Application for Pontianak

Volume 2: Annexes



Master Thesis
J.B.P. van Driel
September 1993

Delft University of Technology
Faculty of Civil Engineering, Hydraulic Engineering Division

HASKONING Royal Dutch Consulting Engineers and Architects
International Projects Division

Table of contents

Annex 1 Detailed description of modelling-procedures

1.1 Cargo-commodities/ feeder cargo

The cargo which is handled at the terminal(s) in the model is divided into four commodities. These commodities are: containers, breakbulk, liquid bulk and dry bulk. The character-attribute *Ct_code* determines the commodity of a cargo-type:

Ct_code:	Commodity:
"A"	Containers
"B"	Breakbulk
"C"	Liquid bulk
"D"	Dry bulk

The reason for making a difference between the four commodities is to take into account the different modes of storage of these commodities.

For containers the specific storage input-and output-characteristics are:
* Cargo flow unit: twenty feet equivalency unit (TEU)
* A separation between export, import and empty containers; each have their own stack
* Separate registration of containers which are transported both to and from the terminal by ship and do not enter or leave the terminal by inland transport (this is discussed further on in this paragraph)
* A separation between 1-TEU and 2-TEU containers
* The storage requirement for the stacks are calculated, based on stackheights in number of containers
* The storage requirement is expressed in occupied number of groundslots and in occupied surface area (m^2)

For breakbulk the specific storage input- and output-characteristics are:
* Cargo flow unit: ton
* A separation between open and covered storage; open storage on the terminal-premises and covered storage in warehouses and sheds respectively
* The storage requirement is calculated, based on the mean stackheight and the relative density
* The storage requirement is expressed in occupied surface area (m^2)

The storage input- and output-characteristics for dry bulk and liquid bulk are identical. They are:
* Cargo flow unit: ton

* The storage requirement is calculated, based on the relative density
* The storage requirement is expressed in occupied storage volume (m^3)

The difference between the commodities arises in the following modules:
* Module *Main* (lines 95-109); in these lines the storage characteristics of each cargo-type are read from User Data File T-file
* Module *Generator* (lines 34-37); in these lines the percentage of forty-feet/2-TEU containers of containers-ships is calculated. The unit of the consignment-size of container-ships is TEU and the unit of the unloading-rate and loading-rate of cranes is box/hour. In order to make a correct calculation of the length of time of unloading and loading a ship, the percentages a fortyfeet-containers is required. The attributes *Ship_fortyfeetperc_imp* and *Ship_fortyfeetperc_exp* are applied in the module Terminal, when the combined unloading-rate and combined loading-rate of all cranes, which are serving one ship, are calculated. The default-values of these attributes for ships with breakbulk, liquid bulk and dry bulk are determined in lines 36 and 37.
* Module *Storage* (lines 52-79); in these lines the different types of calculations of storage-requirements are performed and the different types of output of storage-requirements are stored.
* Module *Report* (lines 24-44); in these lines the throughputs for the different commodities are specified in the Report-file.

The advantage of taking into account the four different commodities is that an extra detail is achieved in the model. By distinguishing the commodities, the facilities of the model for analyzing the performance of different aspects of a port increase. The disadvantage is that an irregularity is created in the input-data of the model. Each commodity has different types of storage-characteristics, increasing the chance of errors in the input. Secondly also irregularities occur for storage-calculations and unloading/loading-rate-calculations.

Another special detail for cargo-handling in the model is the so-called *feeder cargo*. This is defined as cargo which is transported to the terminal by a ship and which also leaves the terminal by ship. Inland transport is not involved. One of these ships could be a feeder-ship, for regional container-transport; for example containers delivered to the port by container-carriers and then distributed to other regional ports by feeder-vessels. In this case the port is a regional hub-port.

Feeder-cargo is simulated in the model by offering the possibility to allocate a percentage of the consignment-size of ships of a shipclass to feeder-cargo. In lines 90-92 and 111-113 of the module *Ship*, in which the registration of unloaded and loaded cargo is performed, this percentage is then not added to c.q. subtracted from the regular storage, but added to c.q. subtracted from a separate feeder-cargo-storage (*Ct_storage_feeder*). In line 113 a check is performed to ensure that the value of *Ct_storage_feeder* does not become negative. In the module Storage, the value of *Ct_storage_feeder* is stored in a storestream; this facility is available for containers.

In addition to cargo carrying vessels, the model also offers the possibility for passengerships to call at the port. For the cargo type *passengers* dummy values must be entered. Passengerships have *Ct_code "E"*. No output information about the passengerflow are given by the model.

1.2 Cargo-arrival-pattern and cargo-departure-pattern

The arrival-pattern of a cargotype describes the dwell time distribution of a cargotype which is exported. It contains a set of percentages, representing the quantity of daily arrivals of cargo with inland transport at the terminal before the arrival of the ship, with which the cargo will be exported. The departure-pattern describes the dwell time distribution of a cargotype which is imported. It contains a set of percentages, representing the quantities of daily departures of cargo with inland transport after the departure of a ship with which it was imported.

The way in which such a pattern is constructed is as follows: the user constructs a matrix of percentages of arrivals/departures on each day of the pattern and for each of the three transport-modes (road/rail/inland water transport). The sum of these percentages is the total percentage of indirectly transhipped cargo. The percentages of cargo that are transhipped directly to and from truck/wagons/barges also have to be specified by the user. Added together the sum of all percentages is 100%.

The user has to define a default value for the general length of all arrival-patterns and all departure-patterns. This value is equal to the respectively the longest arrival-pattern of all cargotypes and the longest departure-pattern of all cargotypes. If a pattern of one cargo-type is shorter than the pattern of another the remaining days should be filled with zero-entries. Besides, the component *Main* automatically adds two zero-entry-days to the arrival-patterns, which represent weekend-days with no inland transport, and for every next five days of the arrival-pattern *Main* adds another two zero-entry-days. All zero-entry days are placed at the beginning of the arrival-pattern; in case a reshuffle takes place for a separate, these zero-entry-days are used. In the module *Main* the macro *Patterns* is called to construct the patterns. This macro reads the matrices and reads the percentages of directly transhipped cargo to and from the three different cargo modes. In the module Ship the macro's *Arrival-pattern* and *Departure-pattern* are called to reshuffle the pattern for the cargo of each ship depending on their time of arrival and departure.

Table A1.1: departure pattern matrix

Day number	Truck perc.	Wagon perc.	Barge perc.	Total perc.
0 (direct)	0	0	0	0
1	0	0	0	0
2	0	0	0	0
3	4	1	0	5
4	8	2	0	10
5	16	4	0	20
6	24	6	0	30
7	16	4	0	20
8	8	2	0	10
9	4	1	0	5
Total	80	20	0	100

There are two reasons for giving the cargo-patterns all cargotypes the same length. The first reason is to create a uniformity in the input-data. The second, and more important, reason is that the length of the arrival-patterns of the cargo of all ships must be identical, in order not to disturb the distribution of inter-arrivaltimes of ships, which the user has specified. A ship is generated at an intervaltime with user-defined mean value. At the moment of generation the ship does not become physically present at the port but first stays in the queue *Arrivingships*. In this period, the export-cargo of the ship arrives at the terminal, according to the arrival-pattern of the ships' cargo. When the ships leaves the queue, it becomes physically present at the port, by joining the waiting row at the anchorage. The mean value of the distribution of these actual moments of arrival of ships of one shipclass may not deviate from the user defined value. Therefore the length of each cargo-arrivalpattern, which is equal to the period of a ship in the queue *Arrivingships*, must be the same for the cargotypes of all ships.

The principle of arrival- and departure patterns is illustrated with the following example. The example concerns a cargo which is imported and whose dwell time distribution is described by Figure A1.1.

Other data are:

* 80% of the inland transport is performed by road
* 20% of the inland transport is performed by rail

* The mean dwell time and the maximum dwell time for road and rail are both equal
* The percentage of directly transhipped cargo (= no intermediate storage on the terminal) is zero

The dwell time distribution is transformed into a departurepattern, which is shown in Table A1.1. The departure-pattern is visualised in Figure A1.2.

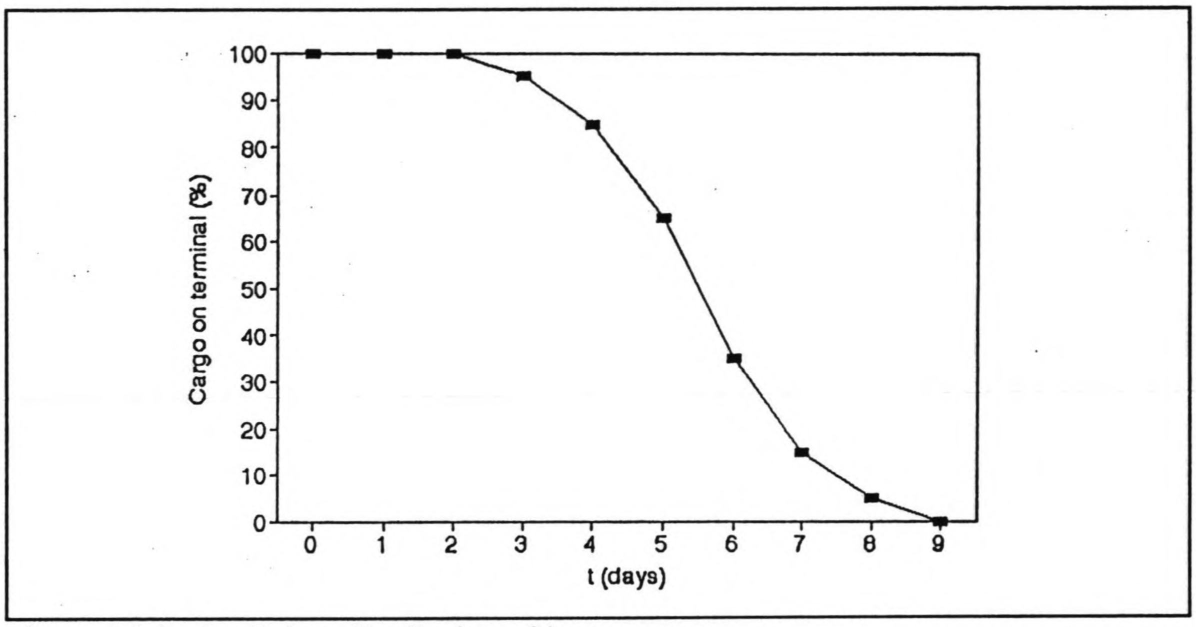


Figure A1.1: dwell time distribution of import-cargo

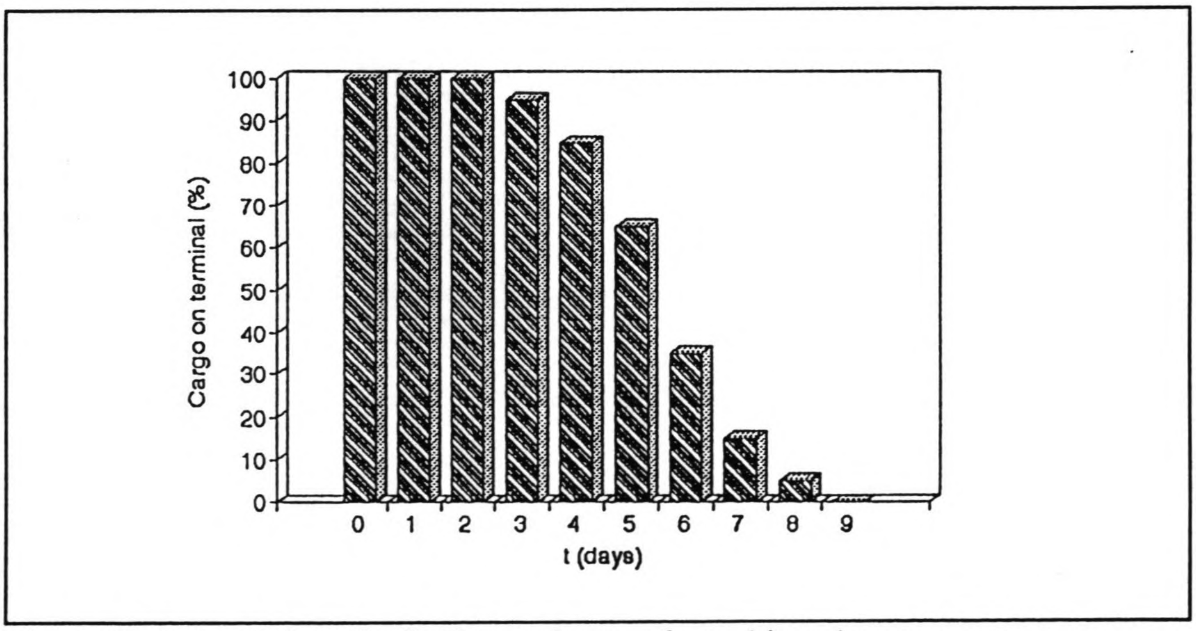


Figure A1.2: dwell time distribution, to be transformed into departure-pattern

Annex 2 Line-to-line description

2.1 Introduction

This chapter contains a detailed description of the program. The program-lines of each module are split up into blocks of lines, which have a uniform purpose or which describe a coherent part of a procedure. For each block the chosen method of programming is discussed.
The module *Define* can be explained as a whole. This module contains the definition of the single components, the class components, the queues, the randomstreams, the inputstreams and the outputstreams. The exact description of these model-accessories are given in the first part of the Lexicon.

The time-unit of the model, which is also defined in this module, is days. This unit is considered to be suitable for port-simulation: besides, it is an appropriate unit for a majority of the input-data. For the other input-data and for the output-data which indicate values of time, the unit hours is used; in the program hours are calculated into days and vice versa.

Thirdly, the module *Define* contains the definition of the attributes of the components. The attributes are of the type real, integer, logical, continuous or they are a reference to a set or to another component. The meaning of each attribute is explained in part II of the Lexicon.

2.2 Module Main

BLOCK 1, MODULE MAIN, LINES 1-7

The program contains ten attributes which are of the type reference to set; seven are attributes of class components and three are attributes of *Main*. In Block 1 the three reference-to-set-attributes of *Main* are initiated. The other seven are initiated in several loops further on in this module.

BLOCK 2, MODULE MAIN, LINES 8-20

In Block 2 the User Data File *C-file*, to which descriptions of errors in the input-files are written, is rewound and headings of this file are written.

BLOCK 3, MODULE MAIN, LINES 21-62

The first inputstream which is read in this module is *H-file*. In Block 3 the data concerning the restrictions bad weather condition, strike and tidal window are extracted from this inputstream. Also the distributions which characterize the first two of these restrictions are shaped, the seeds of these distributions are chosen and the switches of these restrictions are set. Thirdly, the function-description of the water-depth in the entrance channel is specified. This function is as follows:

$$ED = MD + a_1 * \cos\left(\frac{2\pi t}{T_1} - \alpha_1\right) + a_2 * \cos\left(\frac{2\pi t}{T_2} - \alpha_2\right)$$

in which:

ED	=	Water-depth in entrance channel (m.)
MD	=	Mean water-depth in entrance channel (m.)
a	=	Amplitude of tidal component (m.)
T	=	Period of tidal component (days)
α	=	Phase angle of tidal component (rad)

BLOCK 4, MODULE MAIN, LINES 63-77

In Block 4 the general length of the arrival-pattern and the departure-pattern of cargo-types are determined. The arrival-pattern is then prolonged, in order to add weekend-days in the pattern. The default-value for *Satsun* (= the maximum number of saturdays and sundays in the arrival-pattern) is 2. For every extra five days in the pattern, SATSUN is raised by 2. When *Satsun* has been calculated, the length of the arrival-pattern is raised by *Satsun*.

BLOCK 5, MODULE MAIN, LINES 78-87

Block 5 indicates the start of a loop in which each terminal is created. Two class components and three reference-to-set-attributes of the class component *Terminal* are initiated.

BLOCK 6, MODULE MAIN, LINES 88-128

Block 6 represents a loop in which for each terminal the cargo-types which are handled at that terminal are created. The cargo-type-characteristics consist of general information (name, reference-number), the arrival- and departure-pattern and the cargo-storage-data; they are extracted from the inputstream *T-file*. The patterns are read from *T-file* in macro *Patterns*, which is discussed further on in this chapter. Some cargo-storage-characteristics are uniform for each cargo-type, others depend on the commodity of the cargo-type (A for container, B for breakbulk, C for liquid bulk, D for dry bulk, E for passengers). The number of commodity-dependant data varies for each commodity, so a special error-recognition-routine has been included in the loop (lines 117-126).

BLOCK 7, MODULE MAIN, LINES 129-153

Block 7 represents a loop in which for each terminal the berths, which are situated at that terminal, are created. The data are read from *T-file*. The set *Berth_ships* is initiated as an *autostoring set with statistics* because the number of ships in this set is used as output-information. When the harbour-master checks berths for a place for a ship, he starts with the first ship in the suitable set *Term_berthset*, the takes the second, then the third etc. In order to the find the berth with smallest discrepancy between water-depth at the berth and ship-draught, the berths are joined to the *Term_berthset*, ranked by the water-depth at the berth.

BLOCK 8, MODULE MAIN, LINES 153-171

The loop in Block 8 is responsible for creating the cranes
at each terminal. The data are read from *T-file*. Cranes are joined to two sets, one representing all cranes in the model (*Craneset*), one representing all cranes at one berth (*Berth_cranes*). If the berth is a single berth, the cranes are ranked to *Berth_cranes* by the sum of the loadcapacity and unloadcapacity of the crane, so that the terminal-master allocates the cranes with the largest capacity to the ships. If the berth is a multiple berth, the cranes are ranked to *Berth_cranes* by their reference-number, so that the cranes are allocated to the ships in order of physical appearance (the crane-numbers are allocated to the cranes in order of physical appearance).

BLOCK 9, MODULE MAIN, LINES 172-203

The additional information, which is required to characterize each terminal, is extracted from *T-file* in Block 9. It concerns the availability of shifts and inland transport in weekends, the storage-capacity, the terminal-equipment and the shifts. Each shift is activated from this loop.

BLOCK 10, MODULE MAIN, LINES 204-227

The general information for all terminals, which is the last set of data in *T-file*, is read in Block 10. It concerns the data for the maintenance and the breakdowns of cranes. The distributions which characterize the breakdowns are shaped; for each crane in the model a first breakdown-time is drawn from the distribution of inter-arrivaltimes of crane_breakdowns and other default-values are set. Finally, a uniform distribution is shaped, which is required in the module *Generator.*

BLOCK 11, MODULE MAIN, LINES 228-301

Block 11 represents a loop in which for each shipclass the
characteristics of that class are created, by reading the required information from the inputstream *S-file*. A set of seven distributions, characterizing each class, is shaped and their respective seeds are created. In line 286-290 an array of cumulative percentages of terminals for which a ship of the relating shipclass is destined to sail (*Class_term_tab*) is produced. For each shipclass a generator is created and that generator is activated. In line 299 macro *Dwt_tables* is called; in this macro, which is discussed in detail in Paragraph 2.21 of this chapter, the inputstream *D-file* is read.

BLOCK 12, MODULE MAIN, LINES 302-377

In Block 12 the report of errors (*C-file*) is completed. The total number of errors in the input-information is displayed on the screen; if this number is greater than zero, the user is requested to check the *C-file* for a specification of the errors and to correct them. The user is also asked to check the *C-file* for possible mistakes in the input of reference-numbers. The information required for this check is produced in this block. Reference-numbers of cargo-types and cranes, based on the information of the inputstreams *S-file* and *T-file*, are printed in the *C-file*. The program is interrupted by the *Interrupt*-statement in line 376, in order to enable the user to make a choice between continuing the simulation-run, adapting the input-files or checking the *C-file*.

BLOCK 13, MODULE MAIN, LINES 378-389

In Block 13 the single components with an own process description (storage-master, terminal-master, harbour-master, strike, typhoon) are activated. Furthermore, *Main* is instructed to integrate its continuous attribute (the entrance-depth of the harbour) until the simulation-time is over. This implies that *Main* will become current component again at the end of the simulation-time.

BLOCK 14, MODULE MAIN, LINES 390-413

During the simulation, the time at rest and the time at work for cranes, the occupied time and not-occupied time for berths and the time at berth for ships are calculated. When the simulation-time is over, the time-measurements must be completed, to ensure the accuracy of these measurements. This is performed by the lines of Block 14.

BLOCK 15, MODULE MAIN, LINES 414-419

In the last block of this module the reporter is activated, all components are cancelled and *Main* is terminated.

2.3 Module Generator

BLOCK 1, MODULE GENERATOR, LINES 1-12

In lines 7 and 8 an inter-arrivaltime for a ship of the shipclass, of which this generator generates ships, is drawn. Which of these two statements is performed depends on the shipclass; line 8 if it is a shipclass of feeders and a normal distribution is used for the inter-arrivaltime; line 7, if an exponential (Poisson) distribution is used. The generator then waits during the inter-arrivaltime and creates the new ship.

BLOCK 2, MODULE GENERATOR, LINES 13-25

In Block 2, the attributes of a ship are determined. The attributes *Ship_importtotal* and *Ship_exporttotal* are drawn in lines 15 and 16. In line 17 the value for the dead weight tonnage is drawn from a uniform distribution. Based on the DWT-value, the attributes *Ship_draught_berthed* and *Ship_length_berthed* are determined, using the DWT-tables of the particular type of ship. If required for the modelling of a port, *Ship_draught_berthed* can be multiplied by a factor 1.15, representing the underkeel-clearance, and *Ship_length_berthed* can be multiplied by 1.10, representing the margin between ships at the quay. For Pontianak this multiplication is omitted because the respective safety margins are already included in the values for lengths and draughts in the DWT-tables. For the attribute *Ship_cover_fac* the normal distribution needs to be truncated, in order to avoid unreal values. The boundaries are equal to the mean value +/- two times the standard deviation. The truncation is performed by the *While..end*-loops in lines 22-24.

BLOCK 3, MODULE GENERATOR, LINES 26-38

In Block 3 a ship is allocated to a terminal. This is achieved by the following procedure: a random percentage between 0 and 100 is drawn from a uniform distribution (*Rand*). The array of cumulative percentages of terminals to which a ship is sailing (*Class_term_tab*) is then checked (the *For..end*-loop, lines 28-30), starting with the value for the last terminal. If the random number is smaller than the value for a terminal in the array, the attribute *Ship_term_number* is given the value of the reference-number of that terminal. At the end of the loop the value that *Ship_term_number* holds at that moment indicates the terminal to which the ship will sail. Example: with an array *Class_term_tab* as follows:

x	*Class_term_perc*	*Class_term_tab[x]*
1	20	20
2	40	60
3	0	60
4	0	60
5	30	90
6	10	100

and with a value of 56.89 for *Rand*, a ship will be allocated to terminal 2.

NB: At first glance using the *Tabulate*-function and *Value of...at*-function (combined with the *Ceil*-function to make integers of the interpolated values) would seem to be more appropriate in this situation. However, Prosim does not perform this algorithm correctly when there are several identical values. In the above example Prosim would interpolate a value between x=2 and x=5 if 60<*Rand*<90.

BLOCK 4, MODULE GENERATOR, LINES 39-45

The class component *Ship* has one continuous attribute: *Ship_load*. This attribute must be specified but this may happen only once, because for a continuous attribute a specification is valid for all active or later to be created components of that class. The specification of *Ship_load* takes place in block 4. This is however only the case the first time a generator reaches block 4; then *Specified (attribute of Main)* receives the value *True* and lines 39-42 are skipped by all generators during the rest of the simulation. Furthermore in block 4 the ship is activated and the generator is ordered to repeat its process from the start.

2.4 Module Harbour

This module consists of a multiple loop, in which the harbour-master tries to allocate a ship to a berth. When a positive answer has been found, the harbour-master leaves the multiple loop and repeats its process from the start. If no allocation can take place, the harbour-master waits for a ship to leave the port or for a new ship to join the waiting-row at the anchorage. For the sake of clearness the multiple loop has been split up into three blocks.

BLOCK 1, MODULE HARBOUR, LINES 1-13

The harbour-master allocates each ship that enters the waiting-row at the anchorage to a berth inside the port. Therefore the harbour-master is passive while the waiting-row is empty or while all berths are occupied (lines 6 and 7). When neither of these situations is the case, the harbour-master takes a ship from the row (starting with the first, according to the principle of *First In, First Out*): this ship is called *Hm_checkship*. In case that that ship has already been allocated to a berth but is still at the anchorage because it cannot enter the port due to bad weather conditions or to a closed tidal window, it is skipped by the harbour-master (line 13).

BLOCK 2, MODULE HARBOUR, LINES 14-33

The harbour-master takes the first berth from the set of berths of the terminal to which *Hm_checkship* has been allocated; this berth is called *Hm_checkberth* (lines 14-16). In the remaining lines of this block the harbour-master checks if *Hm_checkship* can be allocated to *Hm_checkberth*. Three basic conditions must be met to allow allocating a ship to a berth:

* (1) Three quantitative comparisons must be checked and found positive: free berth-length, water-depth, maximum number of ships (line 18)
* (2) A qualitative check must be performed, to make sure that the ships' cargo is transhipped at the berth (this is discussed further on)
* (3) A ship, which is destined for the same terminal and which has priority to *Hm_checkship*, must not also be in the waiting-row at the anchorage. In line 17 the harbour-master searches for a ship in the waiting-row which fulfils the following conditions:

- The ship has not yet been allocated to a berth
- The ship is allocated to the same terminal as *Hm_checkship*
- The draught of the ship is smaller than the water-depth of *Hm_checkberth*
- The ship belongs to a class which has priority, implying that it is in queue *Priority_row* (FiFo-principle is neglected)

If this ship exists, it is called *Hm_priority_ship*. If the type of cargo which is carried by *Hm_priority_ship* is transhipped at *Hm_checkberth*, *Hm_priority_ship* has priority to *Hm_checkship*. The procedure of finding a berth for *Hm_checkship* can therefore only continue if *Hm_priority_ship* does not exist or if it is the same ship as *Hm_checkship* or if *Hm_priority_ship*'s cargo is not transhipped at *Hm_checkberth*. These three aspects are checked in lines 19-23.

The *For..end*-loop of lines 24-32 is reached if conditions (1) and (3) are fulfilled. In line 25 the harbour-master checks condition (2): is the cargo of *Hm_checkship* transhipped is at *Hm_checkberth* (lines 19-20)?

Résumé of the priority-check: if *Hm_checkship* must give priority to another ship further in the row or if *Hm_checkship* cannot be allocated to *Hm_checkberth* for qualitative or quantitative reasons, this is discovered in lines 18-25. In this case, the harbour-master continues with the next berth at the terminal (line 36) or the next ship in the waiting-row (line 39). A possible ship with priority, further on in the waiting-row, will be found by the harbour-master later in the harbour-masters procedure.

If all checks of lines 18-25 are positive for *Hm_checkship*, this ship can be allocated to *Hm_checkberth*. The allocation is performed in line 26 by joining *Hm_checkship* to the set *Berth_ships* of *Hm_checkberth*. Also the free berth_length of *Hm_checkberth* is diminished. The harbour-master then reactivates *Hm_checkship* before returning to line 5 for repeating its process.

BLOCK 3, MODULE HARBOUR, LINES 34-45

If one or more checks of line 18-25 are negative, the harbour-master repeats the check for the next berth of the appropriate terminal (line 36). If there is no other berth available, the harbour-master takes the next ship (line 39) at the anchorage and repeats the above procedure within the multiple loop. If the harbour-master has passed the multiple loop entirely, none of the ships at the anchorage can be allocated to a berth at that moment of time. The harbour-master then waits for a ship to leave the port or for a new ship to arrive at the waiting-row and repeats its process.

2.5 Module Terminal

The task of the terminal-master is to allocate cranes to a ship. When a ship enters the port and moors at a berth, cranes (or in the case of Pontianak: gangs) are allocated to the ship. The procedure of allocation depends on the available number of cranes at a berth (*supply*) and on the required number of cranes for a ship (*demand*). The terminal-master has two other tasks: one is calculating values for the rates at which a ship is unloaded and loaded (= sum of the unload-capacity and the load-capacity of the allocated cranes) and the other is activating the cranes which are allocated.

BLOCK 1, MODULE TERMINAL, LINES 1-10

Block 1 is the starting point of this module. In this block the terminal-master becomes active (lines 7-9). The terminal-master is activated when the length of queue *Quay* changes (line 8). When the length increases, the arrival of a ship at its berth is implied and the procedure of allocation is started. When the length decreases, the departure of a ship from its berth is implied and the terminal-master restarts its process; this modelling principle has remained from the original module *Terminal*, in which the terminal-master re-allocates cranes when a ship leaves its berths (see Annex 3.5).

BLOCK 2, MODULE TERMINAL, LINES 11-21

When the terminal-master is activated, it is usually to serve only one ship. However, when, after a typhoon, several ships join the queue *Quay* at once, they all need cranes. Therefore in line 12 of Block 2 a *For each..*-loop is started in which for each ship that does not have cranes allocated yet (indicated by the logical attribute *Ship_crane_alloc* having the value *False*), the allocation-procedure is performed. In lines 19-21 the supply of cranes at berth is determined.

BLOCK 3, MODULE TERMINAL, LINES 22-46

Both the supply (Block 2) and demand (user-defined) of cranes are now known. In the *While..end*-loop in Block 3 (lines 24-34) N cranes are allocated to the ship and are activated, in which N stands for the minimum of the demand for cranes of the ship and the supply of cranes at the berth. For each crane the unloading-rate and the loading-rate of the ship are increased by respectively the unload-capacity and the load-capacity of the crane (lines 26 and

27). For containers, the unit of the crane-capacities is boxes/hour, so the capacities are multiplied by a factor to account for 2-TEU containers. The default-value of that factor is 1. After finishing the *While..end*-loop, the logical attribute *Ship_crane_alloc* is given the value ***True*** (line 37). The ship is reactivated and the terminal-master repeats its process from the start. The statements in lines 35 and 36 have been included especially for the Pontianak-model, in order to raise the value of the (un)loading-rates of ships with bagged cargo (with cargo reference code 203) for the minimum production rate improvement scenario. In this scenario, the (un)loading-rates for ships with bagged cargoes differ from ships with general cargo, although they are moored at the same berth.

2.6 Module Crane

The status of a crane is threefold: a crane is either at rest, at work or suffering a breakdown. Maintenance is supposed to take place during rest-periods. The first block of the model accounts for the work-status, the second and third for the breakdown-status and the last for the rest-status.

BLOCK 1, MODULE CRANE, LINES 1-21

A crane is activated by the terminal-master. The starting point of the cranes' process is line 5 in Block 1 of this module. In lines 6-8 the start of the work-status is registered; then the crane reaches the loop of lines 10-21. In this loop the work-status is simulated by the *Wait*-statement in line 15. The crane waits at this statement while six conditions are fulfilled:

* A shift is taking place at the terminal of the berth in question
* Absence of strikes at the terminals
* Absence of bad weather at the port
* The crane is not suffering a breakdown
* The ship to which the crane is allocated has not finished being unloaded/loaded
* The ship which the crane is serving is not being shifted

If one of the first three conditions becomes false, the loop is repeated, but the crane waits at line 12 for that condition to become true again. If during the *Wait*-statement of line 15, the period of time at work has passed the value of the attribute *Crane_breakdown_time* (implicating that the fourth condition is false), the loop is interrupted, the work-status ends and the process continues in line 22 (breakdown-status). If one of the last two conditions becomes false, the loop is interrupted, the work-status ends and the process continues in line 75 (rest-status). The statements in line 11, 13, 14, 16 and 17 are required for registering the delays and the work-periods.

BLOCK 2, MODULE CRANE, LINES 22-50

If the crane is suffering a breakdown, the rate with which the ship, to which the crane is allocated, is unloaded and loaded, decreases. This is performed in line 26-32 of Block 2; if the ship is being unloaded, the attributes *Ship_rate* (which is the attribute in the continuous function *Ship_load* and at that time has the value of *Ship_unloading_rate*) and *Ship_loading_rate* need to be changed; if the ship is being loaded only *Ship_rate* (which at that time has the value of *Ship_loading_rate*) needs to be changed. The crane then waits during a period of time equal to the duration of the breakdown (line 33).

After that period the effect of the breakdown is registered (lines 39-46, 49-50). The delay, due to the crane-breakdown, to the crane and to the operations at the berth, to which the crane belongs, are calculated. Also the delay to the ship-class of the ship, to which the crane is allocated, is calculated. If during the breakdown another ship has taken the place of the original ship to which the crane was allocated, the delay is registered partly to the original (*Crane_prev_ship*, line 35) and partly to the new one (see next block). For the ship-class and the berth, the calculation depends on whether the ship in question is still moored at the berth (*Shipcall2* = 0) or whether it has already left (*Shipcall2* > 0); for the berth it also depends on whether the berth is single or multiple. Also a new time-interval until the next breakdown is drawn *(Crane_breakdown_time*, line 48) and the duration of the next breakdown is drawn (*Crane_breakdown_duration*, line 47).

BLOCK 3, MODULE CRANE, LINES 51-74

The crane-breakdown-status continues in Block 3. If during the breakdown another ship has taken the place of the original ship to which the crane was allocated, the delay due to the breakdown to the new ship is calculated (lines 59-62) and the crane is given new x-coordinates, in accordance with the position of the new ship (lines 56-57). Finally the unloading-rate and the loading-rate of the ship to which the crane is allocated at that moment are increased with respectively the unload-capacity and the load-capacity of the crane. If the ship is still being unloaded/loaded, the crane returns to the work-status (Block 1), otherwise the crane goes to the rest-status (Block 4).

BLOCK 4, MODULE CRANE, LINES 75-80

Block 4 represents the rest-status of the crane. In lines 76-78 the default-values for the rest-period are set; then the crane passivates itself.

2.7 Module Ship

The process of a ship consists of covering a route through several different queues. The division of this module into blocks is based on these queues. The queues are *Arrivingships* (Block 1), *Row* (Block 2), *Port* (Blocks 3-6), *Quay* (Blocks 4 and 5), *Buoy* (Block 7) and *Departingships* (Blocks 6 and 7). Sets which a ship enters are *Berth_ships*, to which a ship is joined by the harbour-master (Block 3-5), and *Term_ships* (Blocks 2-6). The overlap between the queues is demonstrated in Figure A2.1.

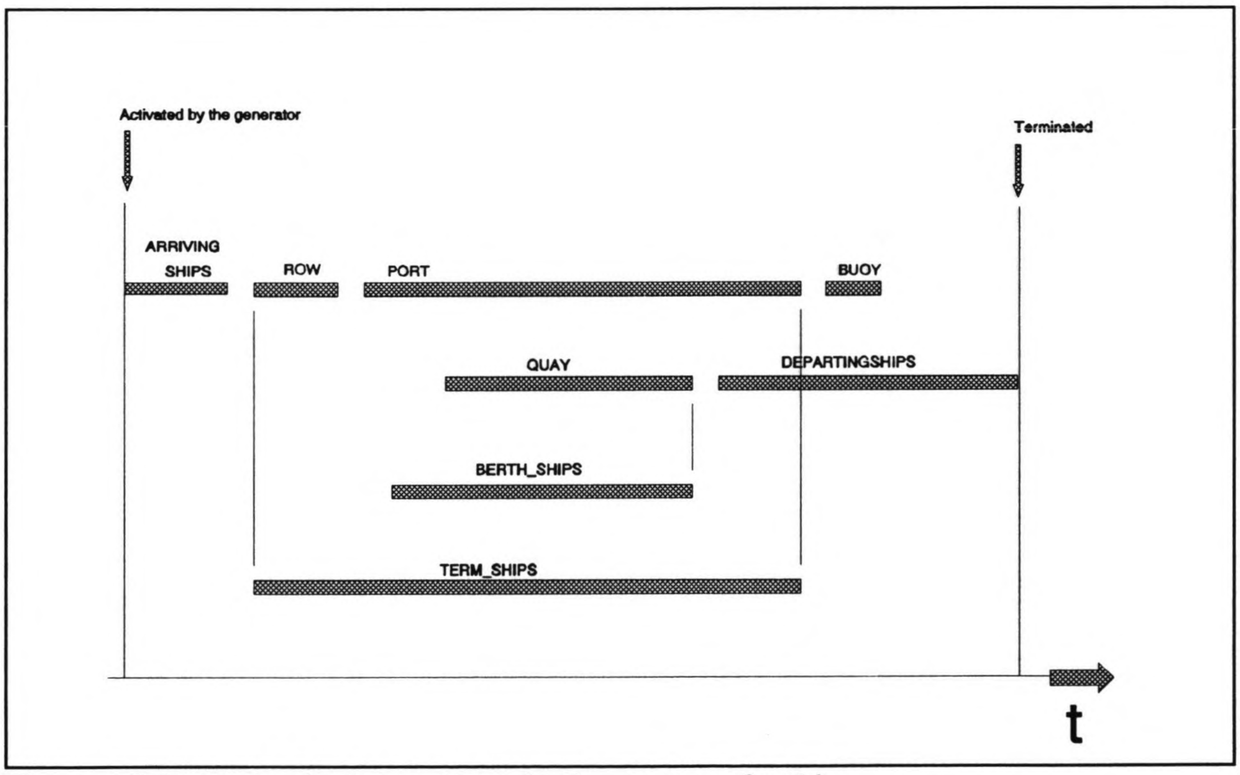


Figure A2.1: overlap between queues in the process of a ship

BLOCK 1, MODULE SHIP, LINES 1-20

When a ship is generated, the first activity of its process is to transform the general arrival-pattern of the cargo-type it is transporting, into a specific pattern for the ships' cargo. This is performed by calling macro *Arrivalpattern* in line 6 of Block 1. This macro is discussed later. If a typhoon or strike is taking place at the moment the ship is generated, the arrival-pattern needs to be changed for the period of time that the typhoon or strike overlaps with the arrival-pattern. This is performed in lines 8-15, in which the macro *Patternchange* is called. This macro is also discussed further on in this chapter. The ship then enters the queue *Arrivingships* (line 16), implying that the export-cargo of the ship is arriving at the terminal. It stays in the queue during a period of time equal to the length of the arrival-pattern (line 17) plus possible extra days, which may be a result of macro *Patternchange* (line 18). This waiting period is split up into two *Wait*-statements, one during the length of *Length_arrivalpatt* and one during the length of *Ship_arr_extra*, because the value of *Ship_arr_extra* can change during the waiting period of *Length_arrivalpatt*. The ship then leaves this queue.

BLOCK 2, MODULE SHIP, LINES 21-44

In line 21 of Block 2 the ship enters the queue *Row*; this means the ship becomes physically present at the port, by joining the row of waiting ships at the anchorage. Ships of a shipclass which should receive priority from the harbour-master also enter the queue *Priority_row*. The arrival of the ship is also registered for the terminal for which it is destined (lines 23 and 24); then the ship passivates itself, in order to be allocated to a berth by the harbour-master. When the harbour-master has chosen a berth for the ship, the ship is re-activated from line 26. Under normal circumstances the ship can then sail from the anchorage to its berth. However, this can be delayed due to:

* Bad weather; the ship waits at the anchorage before entering the port during bad weather conditions (line 28). The delays due to bad weather are registered in lines 31-33.
* Closed tidal window; in line 35 a check is performed to determine if the water-depth of the entrance channel is sufficient for the ship to sail through. This check is performed for the current depth in the channel (*Entrance_depth*) and for the depth in the channel at two later moments of time. This is to avoid an insufficient water-depth when the ship is sailing through the channel; the checks are performed for a quarter and for a half of the mooring time, which approximately coincides with the time that the ship sails through the channel (see Figure A2.2). The mooring-time is the user-defined period of time between leaving the anchorage and mooring at a berth. The delays due to the tide are registered in lines 38-40.

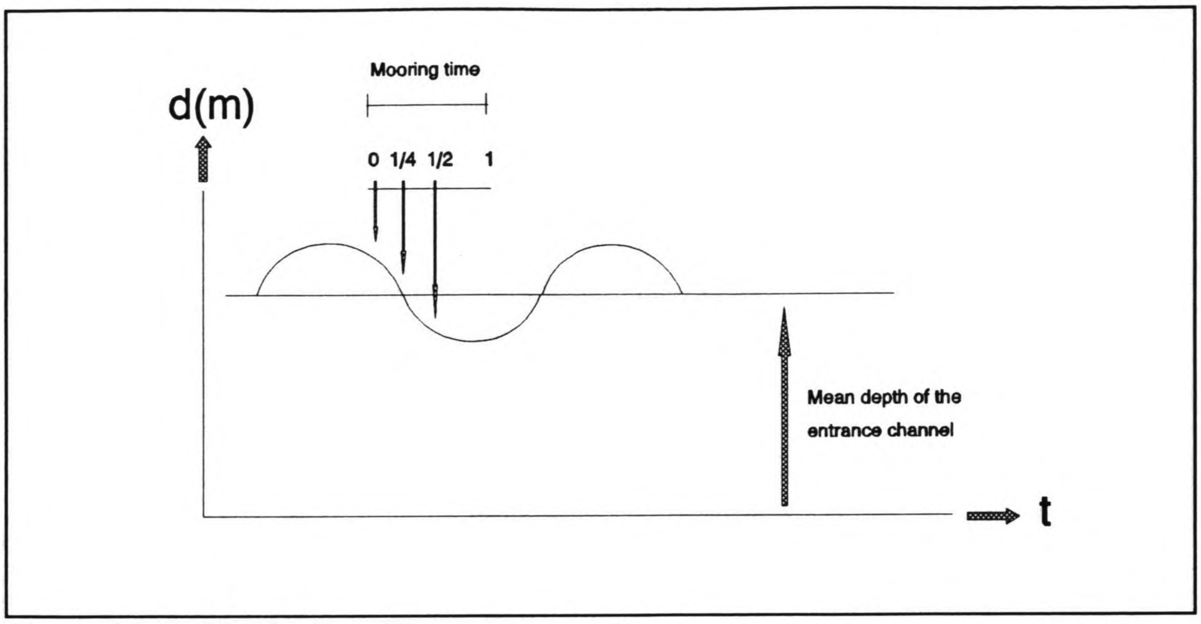


Figure A2.2: tide-check

BLOCK 3, MODULE SHIP, LINES 45-70

Block 3 covers the entry of the ship into the port and the procedure of sailing to a berth and mooring at that berth. The mooring time (period of time for sailing from the anchorage to a berth) has a user-defined value. For a part of the mooring period (the last 0.2 hours) manoeuvring of other ships in front of the berth or at adjacent berths is restricted. During the mooring period a ship can also be obstructed by other ships. The mooring period is divided into waiting during mooring minus 0.2 hours (line 47), waiting due to obstruction by other vessels (line 47) and waiting the last 0.2 hours (line 50). In line 49 the restriction to other ships is created, in line 51 it is cancelled.

After the period of mooring, the ship is ready to have cranes allocated to it by the terminal-master. This is performed by entering queue Quay, which activates the terminal-master. However, this can be delayed if at the time the ship moors at the berth, the terminal-activities are restricted. This can be the case due to:

* Strike; the ship holds its process during a strike (line 54). The delays due to strikes are registered in lines 58-60.
* Bad weather; the ship holds it process during bad weather conditions (line 62). The delays due to bad weather conditions are registered in lines 66-68.

BLOCK 4, MODULE SHIP, LINES 71-93

In the first line of Block 4, the ship enters the queue *Quay*, thereby activating the terminal-master to allocate cranes to the ship. While the terminal-master is doing so, the ship passivates itself (line 74). The ship is re-activated by the terminal-master at line 75. Then the rate at which the ship is unloaded is determined (as the sum of the unload-capacity of the allocated cranes and the capacity of the ships' gear) and the unloading of the ship may start. The unloading-procedure consists of a loop (lines 76-86) which is built up around line 80, in which the attribute *Ship_load* of the ship is integrated. *Ship_load* is the total amount of cargo which has been unloaded from and loaded into a ship. The integration of *Ship_load* can be interrupted by the following circumstances:

* A strike has started
* A bad weather-spell has begun
* The shift has stopped
* All import-cargo has been unloaded

In case of the first three, the loop is repeated; the ship waits in line 78 for the restriction to end. If all import-cargo has been unloaded the loop is interrupted and the unloaded cargo is registered (lines 89-92). The registration of stored import cargo is divided into three categories: import cargo stored under cover, import cargo stored open on the terminal (both of these are transported from the terminal with inland transport) and import cargo which is transhipped via the terminal to another ship (so-called *feeder cargo*). Import cargo which is stored under cover is only of relevance for breakbulk; for other commodities the attribute *Ship_cover_fac* has a default value of zero. Also the total throughput of the cargo-type on the terminal is registered.

BLOCK 5, MODULE SHIP, LINES 94-119

When the unloading is performed, as described in the previous block, the loading of the ship may start. First the rate at which the ship is loaded is determined (as the sum of the load-capacity of the allocated cranes and the capacity of the ships' gear). The loading-procedure is identical to the unloading-procedure and consists of a loop (lines 97-107) which is build up around line 101, in which the attribute *Ship_load* of the ship is integrated. *Ship_load* is the total amount of cargo which has been unloaded from and loaded into a ship. The integration of *Ship_load* can be interrupted by the following circumstances:

* A strike has started
* A bad weather-spell has begun

* The shift has stopped
* All export-cargo has been loaded

In case of the first three, the loop is repeated; the ship waits in line 99 for the restriction to end. If all export-cargo has been loaded the loop is interrupted and the loaded cargo is registered (lines 110-113). The registration of stored export cargo is divided into three categories: export cargo stored under cover, export cargo stored open on the terminal (both of these are transported to the terminal with inland transport) and export cargo which is transhipped via the terminal and has arrived with another ship (so-called *feeder cargo*). Export cargo which is stored under cover is only of relevance for breakbulk; for other commodities the attribute *Ship_cover_fac* has a default value of zero. Also the total throughput of the cargo-type on the terminal is registered. For Pontianak a special procedure for storage of containers is applied. Due to the fact that the cargo which is exported as containerised cargo is stuffed on the terminal premises both the cargo (with cargo reference number 205) and the containers require storage on the terminal. The model only allows one cargotype per shipclass; for this the cargo (205) is chosen. In order to simulate the containerflow a new cargotype with reference number 101 has been created. The flow of containers, which have an average 15 day dwell time on the terminal, is created 15 days before arrival of a ship. This is performed in module *Storage*. In lines 114-117 of this block the containers are removed from the storage

BLOCK 6, MODULE SHIP, LINES 120-156

When the unloading and loading of a ship has finished, (1) the unloaded import-cargo of the ship can leave the terminal and the ship can (2) leave its berth and (3) sail out of the terminal. The first is simulated by entering the queue *Departingships*, the second by leaving the queue *Quay* and the third by leaving the queue *Port*. These actions are described in this block. When a ship enters *Departingships*, first the general departure-pattern of the cargo-type which it is transporting must be transformed into a specific pattern for the ships' cargo. This is performed by calling macro *Departurepattern* in line 121 of Block 6. This macro, which produces a value for the attribute *Ship_extra_dep* (= extension of the departure-pattern for the cargo of this ship) is discussed further on in this chapter.

The ship can then leave the queues *Quay* and *Port* (and the sets *Berth_ships* and *Term_ships*). Leaving *Port* however can be restricted by bad weather conditions and by manoeuvring of other ships. Bad weather conditions are simulated in lines 126-139. The ship waits at its berth during the bad weather in line 128. The delays due to bad weather are registered in lines 136-138. The time required for leaving the berth is equal to the mooring time of a ship (period of time for sailing from the anchorage to a berth). For the first part of that period (the last 0.2 hours) manoeuvring of other ships in front of the berth or at adjacent berths is restricted. During that period a ship can also be obstructed by other ships. The period of leaving is therefore divided into waiting due to obstruction by other vessels (line 135), waiting 0.2 hours (line 154) and waiting during the length of a mooring period minus 0.2 hours (line 155). This last period is performed after leaving queue *Port*, enabling another vessel to be allocated to the berth the passengership has just left. In line 153 the restriction to other ships is created, in line 155 it is cancelled. Before leaving the queue *Port*, several facts concerning the ships' performance are registered (lines 144-151).

BLOCK 7, MODULE SHIP, LINES 157-168

At the start of Block 7, the ship has left the queue *Port* (and has thereby activated the harbour-master to find a new ship for the berth it has left) but it has not yet physically left the port. First the tidal window has to be checked. This is performed by entering the queue *Buoy* and checking the water-depth in the entrance-channel. This check is performed in line 159 for the depth in the channel at two moments when the ship is sailing through the channel. The checks are therefore performed for three quarters of the
mooring time and at the end of the mooring time, which approximately coincides with the time that the ship sails through the channel. The delays due to the tide are registered in line 162.

When the tidal window has opened, the ship leaves the queue *Buoy* and waits during the remaining period of the departure-pattern of its cargo. This waiting period is split up into two *Wait*-statements, one for waiting during the length of *Length_departurepatt* and one for waiting during the length of *Ship_extra_dep*, because the value of *Ship_extra_dep* can change during the waiting period of *Length_departurepatt*. The ship then leaves the queue *Departingships* and terminates itself.

2.8 Module Shift

The objective of this module is to attach the right value to the logical attribute *Term_shift* of each terminal, which indicates whether a shift is taking place or not. This module has been constructed, taking into account that shifts can be cancelled during weekend-days and that the user is free to choose the starting-point of a shift at any hour of the day, even if the start takes place on day x and the finish on day x+1.

On the first day of the simulation the class component *Shift* waits until the starting moment of the first shift (line 6); then the component goes into a loop, which is repeated for each day on which shifts are available. The components holds it process in the loop if in weekends either one (line 9) or two (line 10) days have no shifts. In the daily loop (lines 8-18), a *For..end*-loop is performed (lines 11-17), for each shift; in this loop the component waits during a period of time equal to the duration of the shift (line 13) and then waits during a period of time between two shifts. The latter period depends on whether the last shift of a day has just ended and the next shift starts on the next day (line 16) or whether the next shift starts on the same day (line 15).

2.9 Module Storage

The storage-master is the component which registers the arrivals of cargo at each terminal and the departures of cargo from each terminal with inland transport. This is performed by a daily check of all ships in the queues *Arrivingships* (Block 2) and *Departingships* (Block 3). Secondly the storage-master calculates the consequences of the daily arrival and departures of cargo to the total storage quantities of each cargo-type on each terminal (Blocks 4-8).

BLOCK 1, MODULE STORAGE, LINES 1-18

The storage-master performs its daily activity at the end of a day. Therefore line 7 contains the *Wait*-statement; then (line 8-15), the attributes describing the daily number of cargo-arrivals (export-cargo) and cargo-departures (import-cargo) of each cargo-type (divided in cargo for open and for covered storage) are put to zero. Lines 16-17 determine the day of the week. Line 18 determines whether a strike or bad weather-spell is taking place; if so, no cargo is transported and the storage-master repeats its process the next day.

BLOCK 2, MODULE STORAGE, LINES 19-30

For each ship in the queue *Arrivingships*, the arrival of import-cargo is registered in Block 2. The exact quantity depends on the arrival-pattern of the ships' cargo-type and on the period of days that the ship has already spent in the queue. This period is equal to the attribute *Ship_day*, which is increased in this block by one day for each ship. For weekend-days on which no inland-transport is available, this procedure also takes place. However, the values for the cargo-arrivals of that day have already been put to zero in the macro *Arrival-pattern*. This macro is discussed later on. For Pontianak a special procedure for storage of containers is introduced. Due to the fact that the cargo which is exported as containerised cargo is stuffed on the terminal premises both the cargo (with cargo reference number 205) and the containers require storage on the terminal. The model only allows one cargotype per shipclass; for this the cargo (205) is chosen. In order to simulate the containerflow a new cargotype with reference number 101 has been created. The flow of containers, which have an average 15 day dwell time on the terminal, is created 15 days before arrival of a ship. This is performed in lines 26-29.

BLOCK 3, MODULE STORAGE, LINES 31-41

For each ship in the queue *Departingships*, the departure of export-cargo is registered in Block 3. The exact quantity depends on the departure-pattern of the ships' cargo-type and on the period of days that the ship has already spent in the queue. This period is equal to the attribute *Ship_day*, which is increased in this block by one day for each ship. The procedure is identical to the procedure of Block 2 except for the fact that on weekend-days, the registration of cargo is skipped. The reason for this difference is explained in the paragraph of macro *Departingships*.

BLOCK 4, MODULE STORAGE, LINES 42-51

Blocks 4-6 consists of a loop in which, for each terminal, the consequences for the occupation of the storage area and storage volume, due to the arrivals and departures of cargo on one day, are calculated. In Block 4 first the occupation-ratios for the storage area and storage volume are put to zero for each terminal (lines 44-45); in second place, for each cargo-type on each terminal, the total arrivals and departures of the past day are respectively added to and subtracted from the total storage-amounts (lines 47-50). These amounts are divided into import and export storage and into open and covered storage.

In the following three blocks the total storage amounts are translated into specific storage characteristics for each commodity: containers (Block 5), breakbulk (Block 6), liquid and dry bulk (Block 7). If the first 100 days of the simulation these three blocks are skipped, to avoid a distorted image of the storage characteristics, due to the fact that at the start of the simulation all storage-facilities are empty.

BLOCK 5, MODULE STORAGE, LINES 52-67

In Block 5, the storage characteristics for containers are calculated: first the stack-sizes are counted, divided into stacks for import, export and empties. Based on the stack-sizes and on the mean stackheights, the number of required groundslots is calculated (also divided into the categories import, export and empties). The calculation is as follows:

$$N_n = S_n / r_n$$

in which:

N_n = number of groundslots
S_n = stack size (TEU)
r_n = mean stackheight (m.)
$_n$ = category (import, export, empties)

Based on the number of groundslots, on the gross area-factor (a factor to account for travelling lanes, etc.) and on the average net area-requirement of one groundslot, the required storage area for containers is calculated:

$$O=(N_{import}+N_{export}+N_{empties})*f*F$$

in which:

O = required storage area for containers (m^2)
f = gross area-factor
F = net required area for one groundslot (2.44 * 6.10 m^2)

The data of required slots and required area are stored in a storestream.

BLOCK 6, MODULE STORAGE, LINES 68-74

In Block 6, the storage characteristics for breakbulk are calculated: the required storage area for breakbulk stored open on the terminal and the required storage area for breakbulk stored under cover in sheds or warehouses are both calculated, based on the total storage-amounts, the density, the average stackheight and the gross area-factor (a factor to account for travelling lanes, etc.) of the cargo-type.

$$O_m=\frac{(T_{export,m}+T_{import,m})*f}{h*\rho}$$

in which:

O = required storage area for breakbulk (m^2)
T = storage amount (tons)
f = gross area-factor

h = mean stackheight (m)
ρ = mean density (ton/m^3)
m = open or covered

For each cargo-type the occupied open and covered storage area are stored in a storestream.

BLOCK 7, MODULE STORAGE, LINES 75-79

Liquid bulk and dry bulk have a uniform storage characteristic: the occupied storage volume. This information is calculated (based on the storage amount, the density and a gross area-factor) and stored in a storestream in Block 7. The calculation is as follows:

$$V=\frac{(T_{import}+T_{export})*f}{\rho}$$

in which:

V = required storage volume (m^3)
T = storage amount (tons)
f = gross (volume) factor
ρ = mean density (tons/m^3)

BLOCK 8, MODULE STORAGE, LINES 80-88

In Block 8 some general storage characteristics are calculated and stored. In lines 80-81, for each cargo-type, the total amount of arrivals and the total amount of departures are stored in storestreams. In lines 84-85 the occupation ratios of the storage volume and the storage area of each terminal, which have been calculated at the end of the previous three blocks, are stored in two storestreams. This completes the daily procedure of the storage-master; the storage-master then returns to start.

2.10 Module Strike

When a strike is generated, all activities on the terminal are postponed until the end of the strike: unloading and loading of ships, arrival and departures of cargo with inland transport. Each crane which is unloading/loading and each ship which is moored at a berth (e.g. it is in queue *Quay*) and is being unloaded/loaded is restricted automatically in respectively the module *Crane* and the module *Ship*. The delay due to the strike to these ships is registered in the module *Strike* (Block 3 and 4). In order to cancel the arrival of cargo with inland transport, the arrival-pattern of the cargo of each ship in the queue *Arrivingships* has to be reshuffled (Block 2). The departure of cargo with inland transport is cancelled automatically in the module *Storage*. However, the ships in the queue *Departingships* have to remain in this queue for a longer period, viz. the length of the strike. This is also performed in Block 2.

BLOCK 1, MODULE STRIKE, LINES 1-10

Block 1 is the introductory block of this module, in which an intervaltime between two strikes is drawn. In the procedure described by this block the strike is waiting to commence.

BLOCK 2, MODULE STRIKE, LINES 11-31

When the intervaltime has ended, the restriction of the strike begins. A duration for the strike is drawn and based on the strike-duration, the so-called *Strike_spell*, is calculated. This attribute indicates an integer number of days in which the strike is active, equal to the number of times the storage-master would have registered the arrival of cargo during the period of the strike. Figure A2.3 shows two examples of calculating *Strike_spell*; in both cases its value is 4.

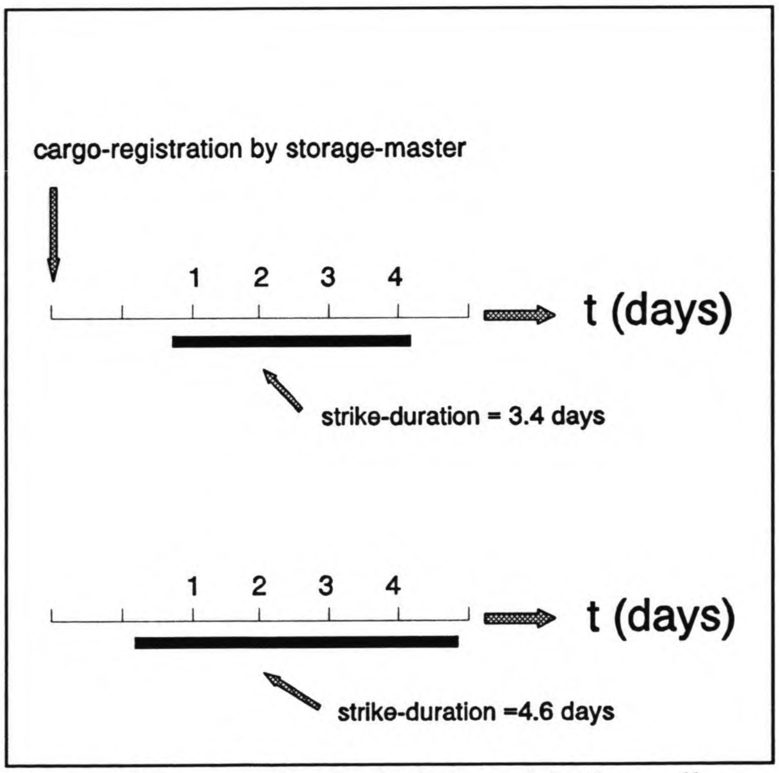

cargo-registration by storage-master
1
2
3
4
t (days)
strike-duration = 3.4 days
1
2
3
4
t (days)
strike-duration =4.6 days

Figure A2.3: example of calculation of *Strike_spell*

Based on the *Strike_spell* the arrival-pattern of the cargo of each ship in the queue *Arrivingships* is reshuffled. This performed by calling macro *Patternchange* (line 16). This macro is discussed further on in this chapter. Next, the length of the departure-pattern of the cargo of each ship in the queue *Departingships* is increased with *Strike_spell*. The consequence of doing so is that for each weekend-day, on which no inland transport is available for one of the cargo-types, the length of the departure-pattern of all cargo of that type has to be increased with an extra day. One day is sufficient because the maximum duration of a strike is five days and a strike can therefore overlap a maximum of one weekend only. The prolonging of the departure-patterns is performed in lines 18-28. The strike then waits during the duration of the strike.

BLOCK 3, MODULE STRIKE, LINES 32-43

In line 33 of Block 3 the number of strikes during the simulation is registered. Also in this block, the delay, due to the strike, to the ships which are moored at a berth, is calculated; this is performed for each ship in the queue *Quay* (lines 39-41). The delay is multiplied by the factor *Term_net_fac* (indicating the net period of time at which shifts are active at a terminal); this implies that periods, at which cargo-handling would not have taken place anyway, are not registered as delay due to a strike.

Table A2.1: overlap of strike and typhoon

Example	Overlap is registered in module...	Delay due to strike (days)	Delay due to typhoon (days)
A	Strike	4	1
B	Strike	4	0
C	Typhoon	1	3

BLOCK 4, MODULE STRIKE, LINES 44-54

The registration of the delay, as indicated in the description of the previous block, has one extraordinary situation, viz. when a strike and a typhoon (bad weather conditions) are active at the same time. In this case the overlap in their active periods are attributed to the restriction which started first. This implies that from the original registration of the delay, in Block 3 of this module or in Block 3 of module *Typhoon*, the overlap-period of the two restrictions must be subtracted from the delay-period,

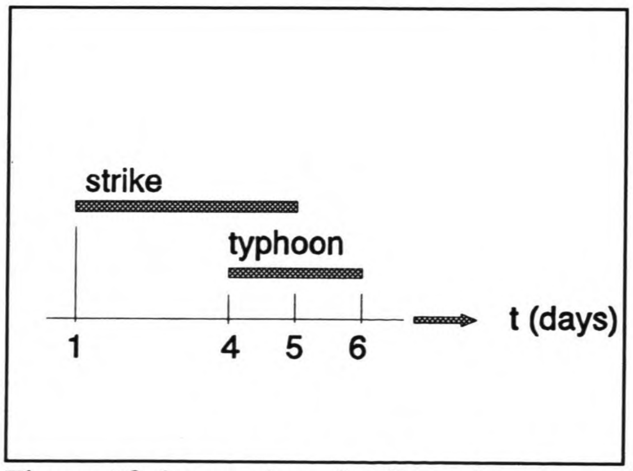


Figure A2.4a: overlap of strike and typhoon

which was attributed to the restriction which started last. This is simulated in lines 49-51 of this module by subtracting the overlap-period from the registration in all cases when the strike started before the typhoon. If the typhoon started before the strike, the substraction of the overlap is performed in the module *Typhoon*. The reason for this procedure of registration is that, if for example the typhoon started first, it is more complicated to subtract the overlap from the strike-delays in the module *Strike* itself than in the module *Typhoon*. In Figures A2.4a/b/c three examples are shown of an overlap between a typhoon and a strike. Table A2.1 shows for each example in which module the overlap is registered (viz. the substraction of the overlap is performed) and how much the eventual contribution to the total delay due to each of the restrictions is.

2.11 Module Typhoon

In this module bad weather conditions are simulated; the word 'typhoon' is used to indicate bad weather conditions. The general principle of the process of a typhoon is nearly identical to the process of the strike. The main difference is that a typhoon has an effect on the terminal-activities and on the manoeuvring of ships in the harbour.

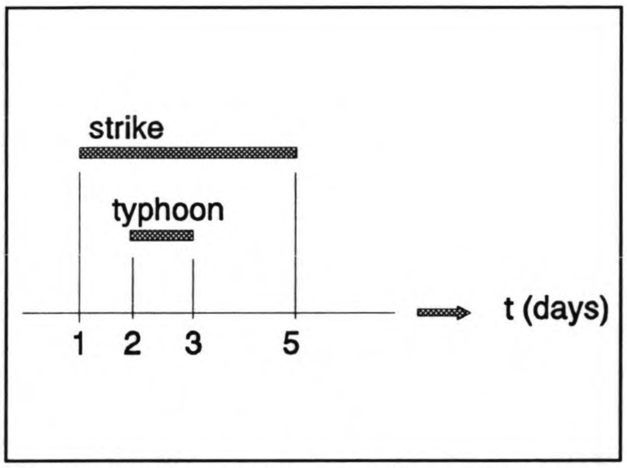


Figure A2.4b: overlap of strike and typhoon

During a typhoon, all activities on the terminal are postponed until the end of the typhoon: unloading and loading of ships, arrival and departures of cargo with inland transport. Each crane which is unloading/loading and each ship which is moored at a berth (e.g. it is in the queue *Quay*) and is being unloaded/loaded is restricted automatically in respectively the module *Crane* and the module *Ship*. The delay due the typhoon to these ships is registered in this module (Block 3 and 4). In order to cancel the arrival of cargo with inland transport, the arrival-pattern of the cargo of each ship in the queue *Arrivingships* has to be reshuffled (Block 2). The departure of cargo with inland transport is cancelled automatically in the module *Storage*. However, the ships in the queue *Departingships* have to remain in this queue for a longer period, viz. the length of

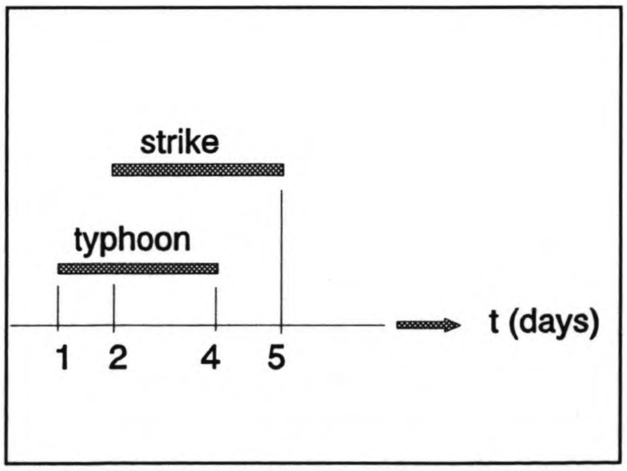


Figure A2.4c: overlap of strike and typhoon

the typhoon. This is also performed in Block 2.

Also during a typhoon, all sailing of ships from the anchorage to a berth and from a berth out of the port is restricted. Checks for bad weather conditions are performed in the module *Ship* each time a ship sails into and out of the harbour.

BLOCK 1, MODULE TYPHOON, LINES 1-12

Block 1 is the introductory block of this module, in which an intervaltime between two typhoons is drawn. In the procedure described by this block the typhoon is waiting to commence (line 10). However before drawing the intervaltime, the component checks if it is the right time of year for bad weather conditions. This is performed by checking if the current time (*Now*) is in between the user defined values for the begin and the end of the typhoon-season (line 8).

BLOCK 2, MODULE TYPHOON, LINES 13-32

When the intervaltime has ended, a duration of the bad weather conditions is drawn. Based on the typhoon-duration, the so-called *Typhoon_spell*, is calculated. This attribute indicates an integer number of days in which the typhoon is active, equal to the number of times the storage-master would have registered the arrival of cargo during the period of the typhoon. The calculation of *Typhoon_spell* is identical to that of *Strike_spell*, as explained in the previous paragraph. Based on the *Typhoon_spell* the arrival-pattern of the cargo of each ship in the queue *Arrivingships* is reshuffled. This performed by calling macro *Patternchange* (line 17). This macro is discussed further on in this chapter. Next, the length of the departure-pattern of the cargo of each ship in the queue *Departingships* is increased with *Typhoon_spell*. The consequence of doing so is that for each weekend-day, on which no inland transport is available for one of the cargo-types, the length of the departure-pattern of all cargo of that type has to be increased with an extra day. One day is sufficient because the maximum duration of a typhoon is five days and a typhoon can therefore overlap a maximum of one weekend only. The prolonging of the departure-patterns is performed in lines 19-29. The typhoon then waits during the duration of the typhoon.

BLOCK 3, MODULE TYPHOON, LINES 33-42

In line 34 of Block 3 the number of typhoons during the simulation is registered. Also in this block, the delay, due to the typhoon, to the ships which are moored at a berth, is calculated;

this is performed for each ship in the queue *Quay* (lines 39-41). The delay is multiplied by the factor *Term_net_fac* (indicating the net period of time at which shifts are active at a terminal); this implies that periods, at which cargo-handling would not have taken place anyway, are not registered as delay due to a typhoon.

BLOCK 4, MODULE TYPHOON, LINES 43-54

The registration of the delay, as indicated in the description of the previous module, has one extraordinary situation, viz. when a typhoon and a strike are active at the same time. In this case the overlap in their active periods are attributed to the restriction which started first. This is simulated in lines 48-50 by subtracting the overlap-period from the registration in all cases when the typhoon started before the strike. This procedure is described in the paragraph of module *Strike*.

2.12 Module Report

The output-file which is produced by the component *Reporter* consists of the following sets of information:

* Performance of the terminals
  - Introduction, number of ships (Block 1)
  - Throughput (containers - Block 2; breakbulk - Block 3; liquid bulk and dry bulk - Block 4)
  - Inland transport (Blocks 5 and 6)
  - Berths (introduction - Block 7; single berths - Block 8; multiple berths - Block 9)
  - Cranes (Block 10)

* Performance of the shipclasses (Block 11)

* Detailed performance of berths (single berths - Block 12; multiple berths - Block 13)

* Indication for performance of terminal-equipment (Block 14)

* Detailed performance of shipclasses (Block 15)

* Performance of restrictions (Block 16)

BLOCK 1, MODULE REPORT, LINES 1-21

In Block 1 the outputstream *Report* is rewound and a heading is made for the file. In line 10 the *For each..*-loop is started, in which the results of each terminal are calculated and printed. The loop ends in line 157 (Block 10). In lines 12-16 of Block 1, a heading is made for each terminal and in line 19 the annual average number of ships which attends a terminal is written to the output-file. The calculation of this number is based on a subtraction of the length of the general cargo-arrival-pattern from the simulation-time, to account for the irregularity that no ships arrive during the first period of the simulation, equal to the length of the arrival-pattern.

BLOCK 2, MODULE REPORT, LINES 22-32

In line 23 of Block 2, a *For each..*-loop is started, in which for each cargo-type on a terminal the throughput (Blocks 2/3/4) and the effort of the inland transport modes (Block 5) are calculated. The loop ends in line 52 (Block 5). In line 25 of Block 2 the annual average throughput of containers (unit: TEU) is calculated and written to the output-file. In this calculation, 100 days are subtracted from the simulation, because the registration of throughput starts on day 101. Lines 26-31 contain messages to direct the user to the store-streams for more specified data.

BLOCK 3, MODULE REPORT, LINES 33-39

In line 34 of Block 3 the annual average throughput of breakbulk (unit: tons) is calculated and written to the output-file. In this calculation, 100 days are subtracted from the simulation, because the registration of throughput is started on day 101. Lines 35-38 contain messages to direct the user to the storestreams for more specified data.

BLOCK 4, MODULE REPORT, LINES 40-44

In line 41 of Block 4 the annual average throughput of dry and liquid bulk (unit: tons) is calculated and written to the output-file. In this calculation, 100 days are subtracted from the simulation, because the registration of throughput is started on day 101. For liquid and dry bulk the way of representation is identical. Lines 42 and 43 contain messages to direct the user to the storestreams for more specified data.

BLOCK 5, MODULE REPORT, LINES 45-57

For each cargo-type at a terminal, the number of trucks, wagons and barges, required for delivering and collecting cargo are calculated in the lines of this block. In these calculations, 100 days are subtracted from the simulation, because the registration of throughput is started on day 101. Lines 53-56 contain messages to direct the user to the right storestreams for data concerning the storage-occupation-rates.

BLOCK 6, MODULE REPORT, LINES 58-65

Based on the calculations in Block 5 in the *For each..*-loop of Blocks 2-5, in Block 6 the results for the annual average occupation of inland transport modes (viz. the number of trucks, barges and wagons leaving and entering the terminal) are written to the output-file.

BLOCK 7, MODULE REPORT, LINES 66-85

In Blocks 7-10 the general performance of the berths and the performance of their cranes are calculated and written to the output-file. First the simulation-time is decreased by the length of the general cargo-arrival-pattern (line 66), for the same reason as indicated in the description of Block 1 of this module. In line 68 a *For each..*-loop is started, in which each berth of a terminal is dealt with. The loop ends in line 125.

The applied procedure for registering delays (in for instance the modules *Ship* and *Crane*) causes a small complication in achieving an aesthetically correct output. If, for example, in the following procedure:

```
SHIPCALL1 = NOW
WAIT WHILE STRIKE_ALARM = TRUE
BERTH_DELAY_STRIKE = BERTHDELAY_STRIKE + NOW - SHIPCALL1
```

no strikes occur, the end-value, which is calculated by Prosim for *Berth_delay_strike*, will not be exactly zero but it will be in the order of 10^{-3} to 10^{-4} or smaller. To avoid this being printed in the output-file, possibly causing the user to doubt the validity of the results, the lines 69-74 of Block 7 are applied, in which 6 attributes of a berth with absolute values of $0.5*10^{-4}$ or smaller are put to zero.

In lines 75-83, the values of 9 attributes of a berth are transformed to values on an annual basis. In line 85 a heading is written for the output of the performance of a berth.

BLOCK 8, MODULE REPORT, LINES 86-95

The annual occupation-time of a single berth, its division in time in full operation, partial operation and not in operation, and the time a berth is not occupied are produced in lines 89-94 of Block 8. The unit for these data is hours, because it is a single berth. Also the annual occupation-rate is written to the output-file (line 88).

BLOCK 9, MODULE REPORT, LINES 96-110

The annual occupation-time of a multiple berth, its distribution in time of full operation, partial operation and not in operation, and the time a berth is not occupied are produced in lines 99-104 of Block 9. The unit for these data is meter-days, because it is a multiple berth. Also the annual occupation-rate is written to the output-file (line 97) and the total number of annually available meter-days (line 105). Line 107 directs the user to the storestreams for additional output-data of multiple berths and lines 108-109 produce the average annual number of shiftings of ships and cranes at multiple berths.

BLOCK 10, MODULE REPORT, LINES 111-129

In Block 10 the performance of cranes at each berth are calculated and written to the output-file; this is performed in a *For each..*-loop (lines 113-124). In lines 114-117 four attributes of a crane are put to zero; the reason for this is discussed in the description of Block 7. In lines 118-123 the division of the annual time-expenditure (in operation, not in operation due to breakdown or delay, in maintenance, at rest) is produced. In lines 124, 125 and 128 the *For each..*-loops of respectively the cranes, berth and terminals are ended and in line 127 the simulation-time is increased to its old value, to account for the decrease in line 66.

BLOCK 11, MODULE REPORT, LINES 130-161

Block 11 describes the general performance of the ship-classes in the model. In line 132 a heading is written to the output-file and in line 135 a *For each..*-loop is started, in which for each ship-class the results are produced. The loop ends in line 160 of this block. In lines 136-142 seven attributes of the component *Shipclass* are put to zero; the reason for this is discussed in the description of Block 7. Next, a heading is made for each class of ships (lines 144-147) and a set of output-data is produced (lines 148-157), including the average time at the anchorage and the average time at the quay (divided in the average length of time in operation, in partial operation and not in operation). The unit is hours.

BLOCK 12, MODULE REPORT, LINES 162-183

In Blocks 12 and 13 information concerning the performance of the berths is produced. Partly it is a repetition of data of Blocks 8 and 9, partly it is more detailed information. In line 162 of Block 12 a heading is written to the output-file and in line 164 the simulation-time is decreased by the length of the general cargo-arrival-pattern (for the same reason as in line 66 of Block 7). In lines 166 and 167 two *For each..*-loops are started; in the first each terminal is called upon, in the second each berth of that terminal is called upon. The loops end in line 201 and line 202 respectively. For the remaining part, Block 12 concerns single berths. The detailed information concerns the distribution of the time in which no operations take place at a berth due to delays and of the time in which the berth is not occupied. The unit is hours.

BLOCK 13, MODULE REPORT, LINES 184-204

Block 13 concerns the performance of multiple berths. As with the previous block, the detailed information concerns the distribution of the time in which no operations take place at a berth due to delays and of the time in which the berth is not occupied. The unit is quay-meters. In lines 201 and 202 the loops which commenced in lines 166 and 167 are ended.

BLOCK 14, MODULE REPORT, LINES 205-220

One part of the output of the model concerns an indication for the occupation-time of terminal-equipment. This output-information is calculated and written to the output-file in Block 14. In lines 206-209, a heading is written. In lines 210-218, for each terminal the annual average number of operating hours for terminal-equipment is calculated. The calculation is based on the sum of the throughputs of all cargo-types on a terminal. The unit for the sum is tons, so for containers an average weight of 10 tons is applied. The sum is then multiplied by a factor, indicating the percentage of cargo which is actually handled by the equipment, and divided by the total capacity of the terminal-equipment. This results in the average annual number of operating hours (line 216).

BLOCK 15, MODULE REPORT, LINES 221-245

In the identical way that the performance of the berths is specified in detail, also the performance of the shipclasses is written down in greater detail to the output-file. This is performed in Block 15, in a *For each..*-loop, producing data for each shipclass. Partly it is a repetition of data of Blocks 8 and 9, partly it is more detailed information; the more detailed information concerns the distribution of the waiting-time at the anchorage and of the time at berth in which no operations take place due to delays. The unit is hours.

BLOCK 16, MODULE REPORT, LINES 246-252

Block 16 writes the average annual of strikes and typhoons to the output-file.

2.13 Module Passengership

The process of a passengership is partly the same as the process of a cargo carrying vessel. The main differences are that passengerships does not enter the queues *Arrivingships* and *Departingships* and that the procedure of unloading/loading is exchanged to a period of (dis)embarkation of passengers. This module has been created especially for the application of the model for Pontianak; therefore the tidal window restriction has also been cancelled.

BLOCK 1, MODULE PASSENGERSHIP, LINES 1-22

When a passengership is generated, it first waits during a period of time equal to the length of the general arrival-pattern (line 7 of Block 1). It does not join the queue *Arrivingships* because it does not have any cargo to carry. In line 8 the passengership enters the queue *Row*; this means the ship becomes physically present at the port, by joining the row of waiting ships at the anchorage. A passengership also enters the queue *Priority_row* because it must receive priority by the harbour-master. Then the passengership passivates itself (line 12), in order to be allocated to a berth by the harbour-master. When the harbour-master has chosen a berth for the passengership, it is re-activated from line 13. Under normal circumstances the passengership can then sail from the anchorage to its berth. However, this can be delayed due to bad weather (line 14).

BLOCK 2, MODULE PASSENGERSHIP, LINES 23-36

Block 3 covers the entry of the passengership into the port, the procedure of sailing to a berth and mooring at that berth and the procedure of (dis)embarkation of passengers. The mooring time (period of time for sailing from the anchorage to a berth) has a user-defined value. For a part of the mooring period (the last 0.2 hours) manoeuvring of other ships in front of the berth or at adjacent berths is restricted. During the mooring period a passengership can also be obstructed by other ships. The mooring period is divided into waiting during mooring minus 0.2 hours (line 24), waiting due to obstruction by other vessels (line 25) and waiting the last 0.2 hours (line 28). In line 27 the restriction to other ships is created, in line 29 it is cancelled.

When the passengership has moored, it enters queue *Quay*. It does not need any cranes allocated. The passengers may now (dis)embark. This takes 5 hours (line 34). The passengership then leaves queue *Quay* and is ready to leave the port.

BLOCK 3, MODULE PASSENGERSHIP, LINES 37-68

Leaving the port is symbolized by leaving queue *Port*. This can be restricted by bad weather conditions and by manoeuvring of other ships. Bad weather conditions are simulated in lines 38-51. The passengership waits at its berth during the bad weather in line 40. The time required for leaving the berth is equal to the mooring time of a passengership (period of time for sailing from the anchorage to a berth). For the first part of that period (the last 0.2 hours) manoeuvring of other ships in front of the berth or at adjacent berths is restricted. During that period a passengership can also be obstructed by other ships. The period of leaving is therefore divided into waiting due to obstruction by other vessels (line 47), waiting 0.2 hours (line 64) and waiting during the length of a mooring period minus 0.2 hours (line 67). This last period is performed after leaving queue *Port*, enabling another vessel to be allocated to the berth the passengership has just left. Several facts concerning the passengerships performance are registered (lines 55-61). The passengership doe not have to join the queue *Departingships* because it does not have any cargo to carry. In line 68 the passengership terminates itself.

2.14 Macro Patterns

Macro *Patterns* is a macro which is called from module *Main* for each cargo-type which is created. In this macro, the general arrival-pattern (Block 1) and departure-pattern (Block 2) of the cargo-type are read from the inputstream *T-file*. The reason for creating a separate macro for this procedure is to clarify the structure of the module *Main*.

BLOCK 1, MACRO PATTERNS, LINES 1-21

In Block 1 the arrival-pattern of a cargo-type is read. In the *For..end*-loop of lines 7-13, the percentage of arrivals of each day of the pattern is read from the *T-file* and is attributed to the newly created class component *Arrivalday*. Also each *Arrivalday* is given a serial-number. In lines 14-20 the percentages of direct arrivals (meaning no intermediate storage on the terminal) and of total arrivals by road, rail and inland waterways are read and checked for errors (the sums of percentages of arrivals must add up to 100 percent).

BLOCK 2, MACRO PATTERNS, LINES 22-37

In Block 2 the departure-pattern of a cargo-type is read. The procedure is basically the same as the procedure for arrival-patterns (Block 1). In the *For..end*-loop of lines 24-30, the percentage of departures of each day of the pattern is read from the *T-file* and is attributed to the newly created class component *Departureday*. Also each *Departureday* is given a serial-number. In line 22 the percentage of direct departures (meaning no intermediate storage on the terminal) is read and in lines 32-34 the percentages of total departures by road, rail and inland waterways are read. The respective data are checked for errors in lines 31 and 35 (the sums of percentages of departures must add up to 100 percent).

2.15 Macro Arrivalpattern

This macro is called from the module *Ship*, at each moment a new ship has just been activated. The task of the macro is transform the general arrival-pattern of the ships' cargo-type to a specific arrival-pattern for the ships' cargo. The transformation depends on the day of arrival of the ship and the availability of inland transport on weekend-days.

A general arrival-pattern consists of a user defined number of days. In module *Main*, the length of the arrival-pattern (which is symbolized by the attribute *Length_arrivalpatt*) is increased by the maximum possible amount of weekend days. This implies that for each cargo-type the arrival-pattern has the same length. In the *For each..*-loop in this macro (lines 11-21) the values of the percentage of cargo-arrivals on each day are allocated to the array *Ship_perc_arr[x]* (line 18), starting with the last day of the pattern. If inland transport is available on each day of the week the pattern does not change and during the first couple of days that a ship is in the queue *Arrivingships* the percentage of cargo-arrivals will be zero. For example, if day 0 is the day of arrival of a ship and if the user defines the following general arrival-pattern for the ships cargo-type:

Day:	-7	-6	-5	-4	-3	-2	-1
Percentage:	5	10	20	30	20	10	5

it means that the length of the array *Ship_perc_arr[x]* will be 11 (7 for the arrival-days plus

4 for the maximum possible number of weekend-days). Macro *Arrivalpattern* will create the following array:

x :	1	2	3	4	5	6	7	8	9	10	11
Perc.:	0	0	0	0	5	10	20	30	20	10	5

If inland transport is not available on one or both of the weekend-days, a zero-value is allocated to that corresponding position in the array (lines 13-16). In the example, if inland transport is not available on both weekend-days and if days 2, 3, 9 and 10 are weekend-days, the following array will be created:

x :	1	2	3	4	5	6	7	8	9	10	11
Perc.:	5	0	0	10	20	30	20	10	0	0	5

Résumé: this macro allocates the values of the attribute *Arrday_perc* (the percentage of arriving cargo on one day) of the component *Arrivalday* to a suitable position in the array *Ship_perc_arr[x]*. On a day with no inland transport, a zero-value is allocated; non-used zero-values remain in the first positions of the array.

2.16 Macro Departurepattern

This macro is called from the module *Ship*, at each moment a ship has just been completely unloaded and loaded. The task of the macro is transform the general departure-pattern of the ships' cargo-type to a specific departure-pattern for the ships' cargo. The transformation depends on the day of calling the macro and the availability of inland transport on weekend-days. The transformation is more simple for the departure-pattern than for the arrival-pattern; it merely consists of allocating the values of the attribute *Depday_perc* (the percentage of departing cargo on one day) of the component *Departureday* to the corresponding position in the array *Ship_perc_dep[x]*. This is performed in the *For each..*-loop of lines 9-12. For the departure-patterns, it is not necessary to enter zero-values into the array, as is the case with arrival-patterns, because the departure-patterns do not have to have the same length for the import-cargo of all ships. If during the departure-pattern a weekend-day occurs with no inland transport available, the day is can just be skipped by the storage-master. In the *For each..*-loop of lines 13-17, the number of extra days in the departure-pattern of each ships' import-cargo, due to weekend-days with absence of inland transport, is determined.

2.17 Macro Patternchange

When a strike or typhoon become active, the arrival and departure of cargo to and from the terminal with inland transport is obstructed. For ships which are staying in the queue *Departingships* (implying that the ships' import-cargo is leaving the terminal by inland transport) this means that they have to stay in the queue for a longer period. This is dealt with in the modules *Strike* and *Typhoon*. For ships in the queue *Arrivingships* a problem occurs. In order to maintain the mean inter-arrivaltime of ships, which is created by the user-defined Poisson inter-arrivalpatterns, the ships cannot stay in the queue *Arrivingships* for a longer period. However, all export-cargo of a ship has to arrive and this is not possible during a strike or typhoon. This means that the arrival-pattern of the ships cargo must be reshuffled. This is performed in macro *Patternchange*, which is called from the modules *Strike* and *Typhoon* for ships which are already in the queue *Arrivingships* at the beginning of the typhoon/strike and from the module *Ship* for ships which enter the queue during a typhoon/strike.

In this macro two situations are distinguished: one in which the typhoon-/strike-spell ends before the last day of the arrival-pattern (Block 1), one in which the spell ends on or after the last day of the arrival-pattern (Block 2). The macro has one local parameter (*Spell*), which indicates an integer number of days in which the strike/typhoon is active, equal to the number of times the storage-master would have registered the arrival of cargo during the period of the strike/typhoon.

BLOCK 1, MACRO PATTERNCHANGE, LINES 1-17

After confirming that the end of the strike-/typhoon-spell is before the end of the arrival-pattern (line 10 of Block 1), the sum of the percentages of arrivals of export-cargo of each day during the spell is determined. This sum is then allocated to the first day after the strike-/typhoon-spell (line 12). The percentages of cargo-arrivals during the spell are put to zero. For example, if a three-day strike takes place on days 5, 6 and 7 of an arrival-pattern, indicated by the following array *Ship_perc_arr*:

x :	1	2	3	4	5	6	7	8	9	10	11
Perc.:	0	0	0	0	5	10	20	30	20	10	5

the array is reshuffled to:

x :	1	2	3	4	5	6	7	8	9	10	11
Perc.:	0	0	0	0	0	0	0	65	20	10	5

BLOCK 2, MACRO PATTERNCHANGE, LINES 18-27

The procedure, which is applied in Block 1, cannot be used in exactly the same way when the strike-/typhoon-spell ends on or after the last day of the arrival-pattern (which is checked in line 18 of Block 2). This is solved by prolonging the arrival-pattern of the cargo until the end of the strike-/typhoon-spell and adding one extra day in which the remainder of the export-cargo arrives at the terminal. If, in the example in the description of the previous block, a three-day strike takes place starting on day 10, the array *Ship_perc_arr* is reshuffled as follows:

x :	...	4	5	6	7	8	9	10	11	12	13
Perc.:	...	0	5	10	20	30	20	0	0	0	15

The procedure of Block 2 is in contradiction with the statement in the introduction of this paragraph, saying that the mean inter-arrivaltime of ships should kept intact. However, an exception to this rule is acceptable, due to a very seldom occurrence and a negligible influence.

2.18 Macro Error-1

This macro is called from the *Main*-module when an illegal data-entry in one of the input-files is detected. The macro has two local parameters: *Karakter*, which indicates the attribute for which the wrong entry has been given, and *C*, which points out the type of illegal entry. The four types of are:

* Illegal entry 0 (line 9)
* Illegal entry, not 0 or 1 (line 10)
* An entry which is too small (line 11)
* An entry which is too big (line 12)

2.19 Macro Error-2

This macro is called from the *Main*-module if an error is detected in one of the input-files, when a set of data should add up to 100 but does not. The macro has two local parameters: *Karakter*, which indicates the data-set for which the error has been made, and *Num*, which indicates the reference-number of the component for which the error has been made (shipclass, cargo-type).

2.20 Macro Error-3

This macro is called from the *Main*-module when the *H-file*, *D-file*, *S-file* or *T-file* contain too much data. The macro has one local parameter, *Karakter*, which indicates the file for which the mistake has been made.

2.21 Macro DWT-tables

In this macro, which is called from Main, the DWT-tables are read from the inputstream *D-file*. A DWT-table is a table expressing the relationship between the dead weight tonnage and respectively the draught and length of ships. For each of the four types of ships (containerships, general cargo ships, tankers and bulkcarriers) the DWT/draught-table and a DWT/length-table are read. Also, a user-defined division of types of vessels can be introduced, as is the case for Pontianak. The first (*Tab_draught[x]*) in line 10, the latter (*Tab_length[x]*) in line 16. The tables have user defined lengths. For each table the length is also read from *D-file* (line 6 and line 12). Tables A2.2, A2.3, A2.4 and A2.5 show DWT-tables for each of the four ship-types. The data in these tables are standard ship-size data, corresponding with the Figures 5.1 to 5.4 in Volume I.

Table A2.2

CONTAINERSHIPS		
DWT (*1000)	d (m)	l (m)
5	6.2	105
10	8.3	140
15	9.2	168
20	10.0	190
25	10.5	210
30	11.1	230
35	11.6	248
40	12.1	266
45	12.5	273
50	12.8	276
55	13.2	278
60	13.5	279

Table A2.3

GENERAL CARGO SHIPS		
DWT (*1000)	d (m)	l (m)
5	6.6	110
10	8.4	127
15	9.6	142
20	10.1	156
25	10.3	168
30	10.4	180

Table A2.4

TANKERS		
DWT (*1000)	d (m)	l (m)
50	9.8	260
100	11.5	316
150	12.8	362
200	14.1	400
250	15.0	427
300	15.7	450
350	16.3	465
400	16.7	480
450	16.8	491
500	17.0	500

Table A2.5

BULK CARRIERS		
DWT (*1000)	d (m)	l (m)
25	10.0	165
50	12.4	203
75	14.0	228
100	15.4	249
125	16.6	265
150	17.4	278
175	18.5	289
200	19.3	299
225	20.0	308
250	20.6	315
275	21.1	322
300	21.6	329

Annex 3 Details of model extensions

3.1 Introduction

This annex gives a detailed description of three possible extensions to the model. The three extensions concern the possibility of shifting ships, the possibility of shifting cranes and the manoeuvring of ships through the entrance channel. The first two have effect on the processes of the harbour-master and the terminal-master. Therefore, new modules *Harbour* and *Terminal* are required. The new processes of these two components and the line-to-line description of their modules are described in Paragraphs 3.4 and 3.5 of this annex. The possibility for shifting cranes and ships are based on two modelling principles: the principle of *single* and *multiple* berths and a coordinate-system with which the positions of cranes and ships can be defined. These principles require a special explanation; this is performed in the next two paragraphs. In Paragraph 3.6 the third extension is discussed; this concerns the process of a ship.

3.2 Single and multiple berths

The original model made a difference between so called single berths and multiple berths. A *single berth* is defined as a berth at which a maximum of one ship can moor. The cranes at a single berth can only be used for that specific berth. A *multiple berth* is defined as a quay which can accommodate one or more ships, depending on the length of the ships and the length of the quay. The maximum number of ships at a multiple berth, also called the *capacity* of a multiple berth, has user-defined value, with a maximum of five. The cranes at a multiple berth can be shifted from one ship to another. If necessary ships can also be shifted at a multiple berth to make place for another ship.

The reason for separating single berths and multiple berths is as follows. The two easiest ways of simulating berths are:

* The principle of *one berth-one ship*. The number of berths at a terminal is equal to the capacity of ships at that terminal. All berths in a model can therefore handle a maximum of one ship. In case two adjacent berths are occupied and the sum of the non-occupied length of both berths is sufficient to accommodate a third ship, this ship cannot moor and has to stay at the anchorage.

* The principle of *occupied quay-length*. A quay has a certain length of which one part is occupied and the other part is not. Each time a ship moors at the quay, the length of the ship is added to the occupied length of the quay, each time a ship departs from the quay

the length of the ship is subtracted from the occupied length of the quay. The only check which has to be performed when a ship wishes to moor is to compare the length of the ship to non-occupied length of the quay. The position of ships at the quay is not taken into account.

The advantage of these two quay-modelling-principles are that they are simple and straightforward. The disadvantages is that they are not flexible and not very realistic in case several types of berths exist. These disadvantages can be solved by the principle of single and multiple berths. If modelling principle of one berth-one ship is realistic single berths can be applied (for example jetties for LPG), if the principle of occupied quay-length is more valid, multiple berths can be applied. In the latter case the mooring-position of ships and the position of cranes on the quay is also taken into consideration. This is discussed in the next paragraph.

Another advantage is that the possibility of shifting cranes and ships is created. The shifting of a crane can take place on two occasions:

* If a ship leaves the quay and the cranes, by which is was served, are allocated to adjacent ship
* If two ships are moored adjacently, with one having a sufficient number of cranes and the other having a deficit of two cranes or more. The allocated number of cranes depends on the discrepancy between the supply of cranes (the number of available cranes to serve the ship) and the demand for cranes (the number of cranes required to unload/load a ship, for instance equal to the number of hatches)

Additionally, in both occasions, certain conditions have to be met to shift a crane. They are described in Annex 3.5.

The shifting of a ship takes place in case a ship is allocated to a multiple berth because the non-occupied berth-length is calculated to be sufficient but the largest free space at the berth is not big enough for the ship in question. A free space is defined as the space between a ship and the end of the berth or the space between two ships. The ships at the quay are then shifted until a large free space is created. The exact procedure of shifting a ship is also described in Annex 3.4.

For Pontianak the second quay-modelling principle (the principle of occupied quay-length) is sufficient. Due to the fact that (1) gangs, which do not have fixed positions on the quay, are applied instead of cranes and (2) there are enough gangs to serve the ships, the necessity of shifting cranes is absent. The mooring of ships at the quays is arranged in such a way that shifting of ships does not take place. Local vessels tender at the quay, to which the principle also applies well.

The shifting of ships is performed by the harbour-master and the shifting of cranes is performed by the terminal-master. For Pontianak, the original processes of the harbour-master and the terminal-master (described in the modules *Harbour* and *Terminal*) have therefore been simplified. The simplified procedures are described in Paragraphs 4.6 and 4.7 of Volume 1 in general and in Annex 2.4 and 2.5 in detail. In Paragraphs 3.4 and 3.5 of this annex, the original procedures are explained.

3.3 Coordinates for ships and cranes at multiple berths

The principle of multiple berths at which the position of the cranes on the quay and the mooring-position of ships alongside the quay is taken into account, requires a method of defining those positions. This is performed by imagining an x-axis alongside the quay; the origin is at one end of the quay and the value of x at the other is equal to the length of the multiple berth (see Figure A3.1).

For the position of a ship, two coordinates are defined:

* x_{bow}: x-coordinate, indicating the position of the bow of the ship
* x_{stern}: x-coordinate, indicating the position of the stern of the ship

The ships are always allocated to a multiple berth with $x_{bow} < x_{stern}$. When a ship is allocated to a multiple berth, it is always allocated with a value for x_{bow}, which is as low as possible. This implies that the value of x_{bow} of the newly allocated ship is equal to zero or it is equal to the value of x_{stern} of an already allocated ship.

The position of cranes at a multiple berth is defined by a so-called *range*. This is defined as the part of the quay which cranes can physically reach for serving a ship. It is defined by two x-coordinates:

* x_{min}: the lower boundary of the range
* x_{max}: the upper boundary of the range

When a crane is serving a ship, the lower boundary is equal to x_{bow} of the ship and the upper boundary is equal to x_{stern} of the ship. When a crane is at rest, the lower boundary is equal to x_{stern} of the ship on one side and the upper boundary is equal to x_{bow} of the ship on the other side (or respectively $x_{min} = 0$ and x_{max} = berth-length if one of those ships does not exist).

The x-coordinates of cranes and ships are illustrated by an example in Figure A3.1.

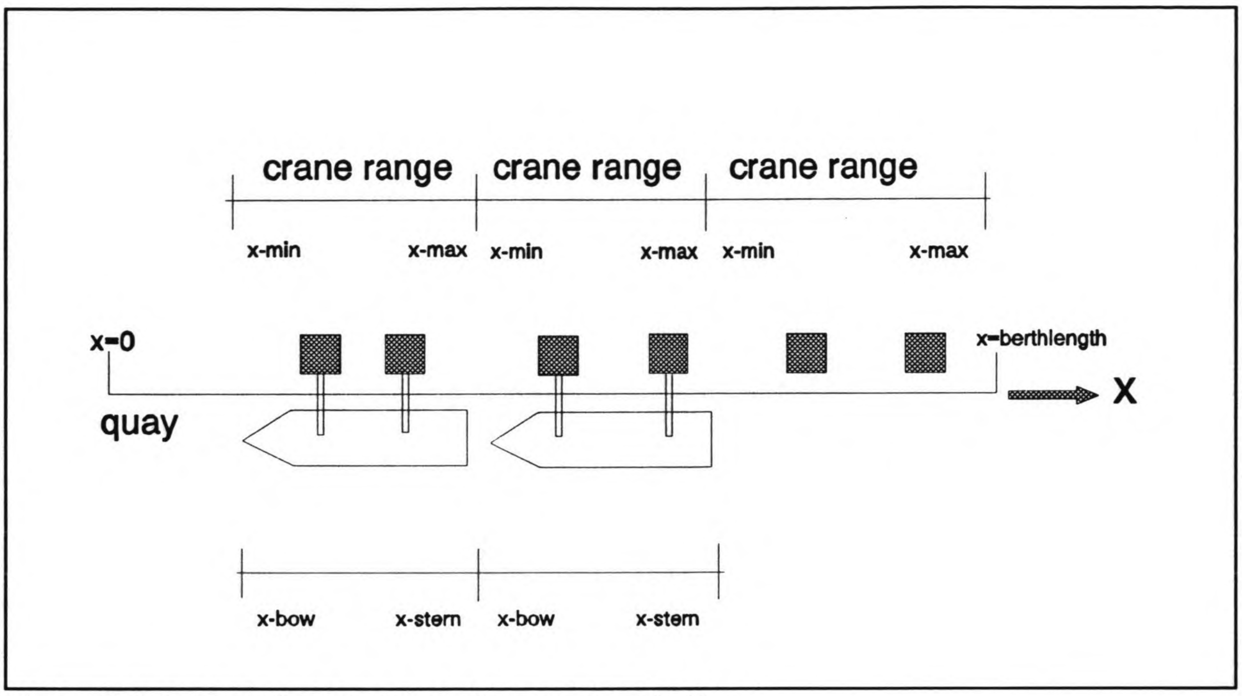


Figure A3.1: Coordinates-system for cranes and ships

3.4 Harbour-master

The task of the harbour-master is to escort ships into the harbour and to allocate them to a berth. A flow chart of the extended harbour-masters procedure is shown in Figure A3.2. The extended procedure is basically identical to the harbour-masters procedure which is described in Paragraph 4.6 of Volume 1. The main difference is the allocation of ships to multiple berths. In the flow chart this is the part of the procedure between allocating a ship to a berth and restarting the procedure. This part of the procedure is now explained.

When the harbour-master has found a ship, for which all checks at a berth are positive, it is allocated to that particular berth. If the berth is a multiple berth, a separate routine is necessary to find a free space at that berth for the ship. It may even be necessary to shift ships at that berth to be able to find enough space. This routine is dealt with in the macro *Shiftships* which is called by the harbour-master.

In this routine the ship is allocated to the smallest possible free space at the berth. The size of a free space is calculated with the use of the a coordinates-system with an x-axis alongside the quay and its origin at one end of the berth. The coordinates-system is explained in the previous paragraph and in Figure A3.1. A free space is defined as the difference between the x-coordinate of the stern of a ship and the x-coordinate of the bow of the next ship, between the zero-point and the x-coordinate of the bow of the first-ship and between the x-coordinate of the stern of the last ship and the total berth-length. If all spaces have insufficient length to accommodate the ship but if the sum of the lengths of the free spaces is sufficient to accommodate the ship, the ships at the quay are shifted. This implies that one by one, starting with the ship at one end of the quay, the ships are moved along the quay, until a space is created which is large enough for the ship to moor.

When this routine is finished the harbour-master reactivates the ship and waits 0.25 hours, to ensure that ships enter the port at a minimum interval of 0.25 hours (see also Paragraph 3.6 of this annex). The harbour-master then repeats its process from the start.

3.4.1 Module Harbour

The extended module is almost identical to the one which is described in Annex 2. Because the line-numbers are different, the whole module is described again in this paragraph. The module consists of a multiple loop, in which the harbour-master tries to allocate a ship to a berth. When a positive answer has been found, the harbour-master leaves the multiple loop and repeats its process from the start. If no allocation can take place, the harbour-master waits for a ship to leave the port or for a new ship to join the waiting-row at the anchorage. For the sake of clearness the multiple loop has been split up into three blocks.

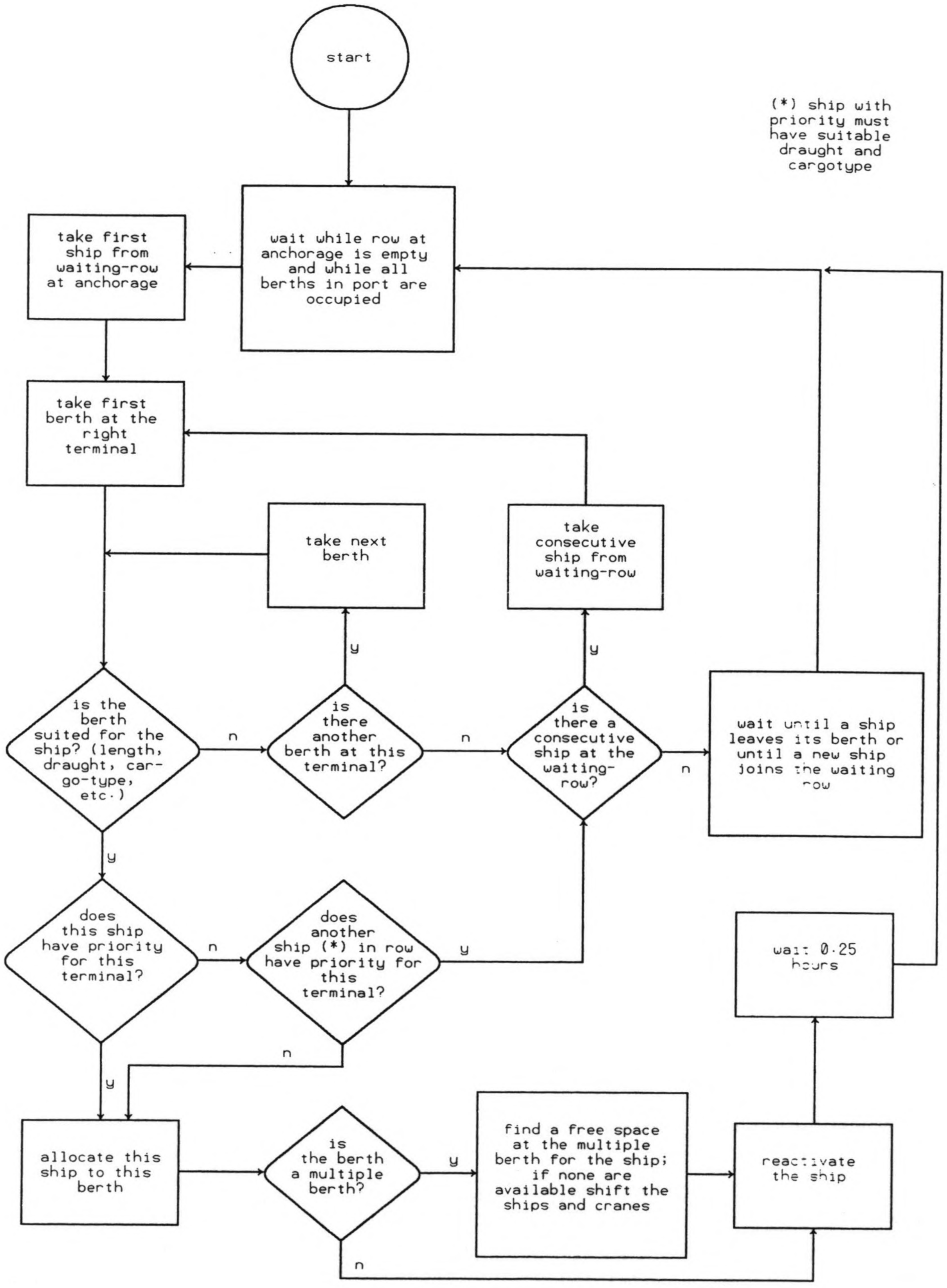
BLOCK 1, MODULE HARBOUR, LINES 1-13

The harbour-master allocates each ship that enters the waiting-row at the anchorage to a berth inside the port. Therefore the harbour-master is passive while the waiting-row is empty or while all berths are occupied (lines 6 and 7). When neither of these situations is the case, the harbour-master takes a ship from the row (starting with the first, according to the principle of *First In, First Out*): this ship is called *Hm_checkship*. In case that that ship has already been allocated to a berth but is still at the anchorage because it cannot enter the port due to bad weather conditions or to a closed tidal window, it is skipped by the harbour-master (line 13).

BLOCK 2, MODULE HARBOUR, LINES 14-40

The harbour-master takes the first berth from the set of berths of the terminal to which *Hm_checkship* has been allocated; this berth is called *Hm_checkberth* (lines 14-16). In the remaining lines of this block the harbour-master checks if *Hm_checkship* can be allocated to

Figure A3.2: extended process of the harbour-master


start
(*) ship with priority must have suitable draught and cargotype
wait while row at anchorage is empty and while all berths in port are occupied
take first ship from waiting-row at anchorage
take first berth at the right terminal
take next berth
take consecutive ship from waiting-row
is the berth suited for the ship? (length, draught, cargo-type, etc.)
n
is there another berth at this terminal?
y
n
is there a consecutive ship at the waiting-row?
y
n
wait until a ship leaves its berth or until a new ship joins the waiting row
y
does this ship have priority for this terminal?
n
does another ship (*) in row have priority for this terminal?
y
n
y
wait 0.25 hours
allocate this ship to this berth
is the berth a multiple berth?
y
find a free space at the multiple berth for the ship; if none are available shift the ships and cranes
reactivate the ship
n

Hm_checkberth. Three basic conditions must be met to allow allocating a ship to a berth:

* (1) Three quantitative comparisons must be checked and found positive: free berth-length, water-depth, maximum number of ships (line 18)
* (2) A qualitative check must be performed, to make sure that the ships' cargo is transhipped at the berth (this is discussed further on)
* (3) A ship, which is destined for the same terminal and which has priority to *Hm_checkship*, must not also be in the waiting-row at the anchorage. In line 17 the harbour-master searches for a ship in the waiting-row which fulfils the following conditions:

  - The ship has not yet been allocated to a berth
  - The ship is allocated to the same terminal as *Hm_checkship*
  - The draught of the ship is smaller than the water-depth of *Hm_checkberth*
  - The ship belongs to a class which has priority, implying that it is in queue *Priority_row* (FiFo-principle is neglected)

  If this ship exists, it is called *Hm_priority_ship*. If the type of cargo which is carried by *Hm_priority_ship* is transhipped at *Hm_checkberth*, *Hm_priority_ship* has priority to *Hm_checkship*. The procedure of finding a berth for *Hm_checkship* can therefore only continue if *Hm_priority_ship* does not exist or if it is the same ship as *Hm_checkship* or if *Hm_priority_ship*'s cargo is not transhipped at *Hm_checkberth*. These three aspects are checked in lines 19-23.

The *For..end*-loop of lines 24-32 is reached if conditions (1) and (3) are fulfilled. In line 25 the harbour-master checks condition (2): is the cargo of *Hm_checkship* transhipped is at *Hm_checkberth* (lines 19-20)?

Résumé of the priority-check: if *Hm_checkship* must give priority to another ship further in the row or if *Hm_checkship* cannot be allocated to *Hm_checkberth* for qualitative or quantitative reasons, this is discovered in lines 18-25. In this case, the harbour-master continues with the next berth at the terminal (line 43) or the next ship in the waiting-row (line 46). A possible ship with priority, further on in the waiting-row, will be found by the harbour-master later in the harbour-masters procedure.

If all checks of lines 18-25 are positive for *Hm_checkship*, this ship can be allocated to *Hm_checkberth*. The allocation is performed in line 32 by joining *Hm_checkship* to the set *Berth_ships* of *Hm_checkberth*. Also the free berth_length of *Hm_checkberth* is diminished. If *Hm_checkberth* is a multiple berth (lines 26-31) the x-coordinates of the bow and of the stern of *Hm_checkship* must be determined. If the berth is empty, the ship is placed at one end of the berth ($x_{bow} = 0$); if the berth is already partly occupied the smallest possible free space has to be found and maybe even the ships have to be shifted. This is performed in the macro *Shiftships*, which will be discussed in the next paragraph. The harbour-master then reactivates *Hm_checkship* and waits 0.25 hours (in order to ensure that there is an interval of 0.25 hours between ships sailing through the entrance channel) before returning to line 5 for repeating its process.

BLOCK 3, MODULE HARBOUR, LINES 41-52

If one or more checks of line 18-25 are negative, the harbour-master repeats the check for the next berth of the appropriate terminal (line 43). If there is no other berth available, the harbour-master takes the next ship (line 46) at the anchorage and repeats the above procedure within the multiple loop. If the harbour-master has passed the multiple loop entirely, none of the ships at the anchorage can be allocated to a berth at that moment of time. The harbour-master then waits for a ship to leave the port or for a new ship to arrive at the waiting-row and repeats its process.

3.4.2 Macro Shiftships

This macro is called from the module Harbour, each time the harbour-master has found a multiple berth, at which sufficient quay-length is free for the ship, that is being checked, to moor. The task of this macro is to find a suitable place at the berth. A suitable place is described at the smallest possible free space at the berth. A free space can be:

* The space between the beginning of the berth and the bow of the first ship
* The space between the stern of one ship and the bow of an adjacent ship
* The space between the stern of the last ship and the end of the berth

The check to find a suitable free space is performed in Block 1. The free spaces are defined by means of x-coordinates (x-axis alongside the quay, with x=0 at the beginning of the berth). If none of the free spaces are large enough for the ship, one or more of the other ships at the quay have to be shifted. This performed by moving the first ship at the berth to beginning of the berth; the check of Block 1 is then performed again. If the created space is not sufficient, the second ship is shifted, right behind the first; the check of Block 1 is then repeated. This routine continues until a space has been found. The shifting of ships takes place in Block 2. Shifting a ship also has consequences for the cranes which are handling the ship in question. This is dealt with in Blocks 3-5.

BLOCK 1, MACRO SHIFTSHIPS, LINES 1-25

In the first lines of Block 1, some introductory statements are made. The attribute *Ship_x_bow* of the ship, which is being checked by the harbour-master, is given value -1; as long as a suitable place has not been found and *Ship_x_bow* remains to have the value -1 and the *While..end*-loop of lines 8-71 is performed. In line 7 the first ship at the berth (= the ship with lowest value of the x-coordinate of its bow) is defined to be *Hm_shiftship*; this is the first ship that will be shifted if necessary. At the end of each *While..end*-loop, the adjacent ship is made *Hm_shiftship*.

Furthermore this block contains the routine to find a suitable place at the multiple berth for the ship which is being checked by the harbour-master; this performed by inspecting the three different kinds of free berth-spaces, as indicated in the introduction of this paragraph. In line 10-13 the space between the beginning of the berth and the bow of the first ship is checked. If the space is large enough, the attribute *Berth_space_free* is given the value of the length of the free space and the attribute *Ship_x_bow* of the ship, which is being is checked by the harbour-master, is given a value, corresponding to the free space. In the *For..each*-loop of lines 14-21 all free spaces between two adjacent ships are checked and in lines 22-25 the space between the last ship and the end of the berth is checked; each time a shorter but still suitable free space is found, *Berth_space_free* and *Ship_x_bow* are given new corresponding values.

BLOCK 2, MACRO SHIFTSHIPS, LINES 26-35

If in Block 1 a suitable free space has not been found for the ship, which is being checked by the harbour-master, the value of *Ship_x_bow* of that ship is still -1. In this case the IF..END-loop of lines 26-70 is performed. In Block 2 the ship at the berth which is currently defined as *Hm_shiftship* is shifted. The ship is moved alongside the quay to a position with x-coordinates, which are as low as possible; this means that if the ship is the first at the quay, x_{bow} becomes equal to zero; otherwise x_{bow} becomes equal to x_{stern} of the adjacent ship. In lines 27-31 of this block new x-coordinates for the bow and stern of *Hm_shiftship* are determined, in lines 32 and 33 *Hm_shiftship* is given a new place in the corresponding set *Berth_ships* and in line 35 the shift is registered.

Figure A3.3 depicts a situation in which a ship (ship 3) requires a space at the quay but in which neither of the three free spaces **a**, **b** and **c** are large enough. First, ship 1 is shifted to the front end of the quay. Because space **a** plus **b** is not large enough either, also ship 2 is shifted. Ship 3 can then more behind ship 2 (see figure A3.4).

BLOCK 3, MACRO SHIFTSHIPS, LINES 36-45

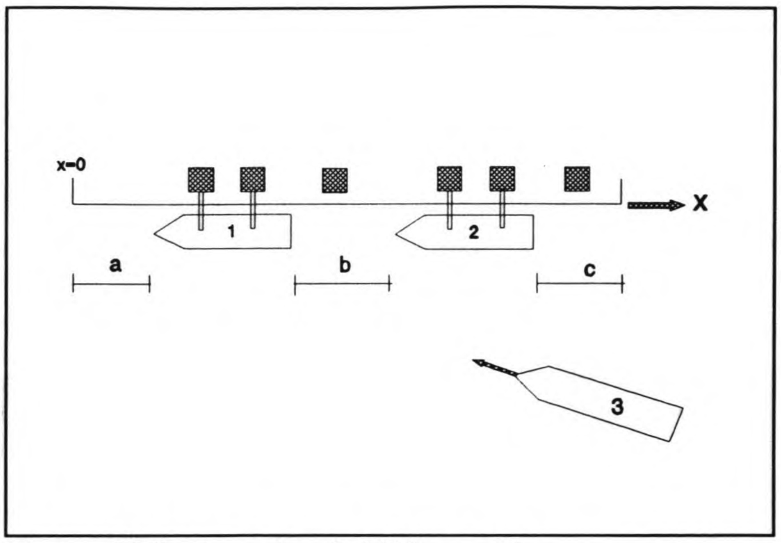


Figure A3.3: example of shifting a ship

In Blocks 3-5 the consequences to the cranes at the berth of shifting a ship are dealt with. If the shifted ship is only using ships' gear, these three blocks can be skipped; this command is performed in line 36 of Block 3. If the shifted ship is being served by cranes, those cranes have to be dismissed and a new set of cranes have to allocated. This is necessary because it is not possible to use the same cranes, which were serving the shifted ship before the shifting, as well after the shifting, without checking if one or more other cranes are standing in between the shifted ship and the adjacent ship in front of the shifted ship. The cranes are therefore put to rest, which is performed automatically by the statement in line 34 of the previous block; in lines 37-43 the x-coordinates of the cranes are changed. In line 44 the crane is searched which will be the one with the lowest x-coordinate to be allocated to the shifted ship. Consequentially this is either a crane which was standing in between the shifted ship and the ship in front of the shifted ship or it is the crane which was also the crane with the lowest x-coordinate, allocated to the ship, before the ship was shifted. This is crane is named *Rightcrane* (line 45), which resembles the crane-allocation-procedure in module Terminal.

BLOCK 4, MACRO SHIFTSHIPS, LINES 46-71

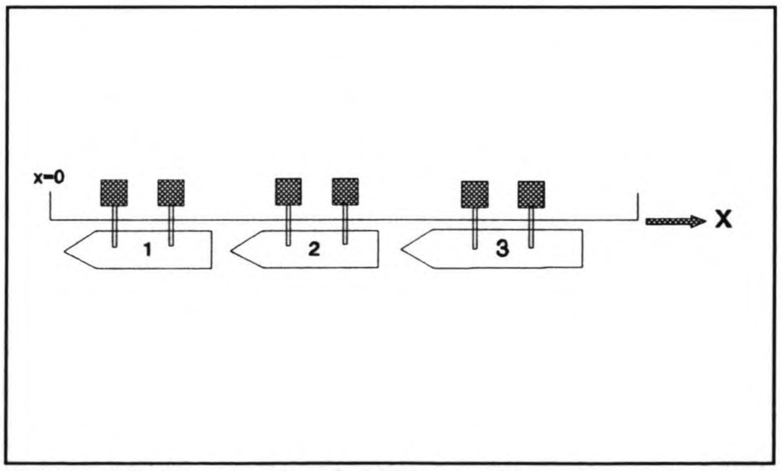


Figure A3.4: example of shifting a ship

Block 4 contains the crane-allocation for the shifted ships and consists of the conclusion of the *While..end*-loop, which started in line 7. In lines 46-48 the unloading/loading- capacity of the ships' gear (if available) is attributed to the ship. In the *For..end*-loop of lines 49-65 a number of cranes is allocated to the shifted ship, equal to the number of cranes which were allocated before the shifting. The allocation-procedure in this loop is identical to the re-allocation-procedure in module Terminal. The difference is that the cranes are not activated straight away but with a delay of one hour (line 57). If the cranes are activated after exactly one hour, the value of *Ship_shifted* of *Hm_shiftship* is still *True*, causing the cranes to be passivated straight away again. To avoid this, the delay in activating the cranes is increased with 0.00001; this has a negligible effect on the simulation-results.

In lines 59-62 a crane which is suffering a breakdown is allocated to the ship in advance, in an identical way as is performed in module Terminal. In line 63, the adjacent crane on the side with higher x-coordinates is picked for the next *For..end*-loop. In lines 67-69 the next ship to possibly be shifted in the next *While..end*-loop is picked.

BLOCK 5, MACRO SHIFTSHIPS, LINES 72-81

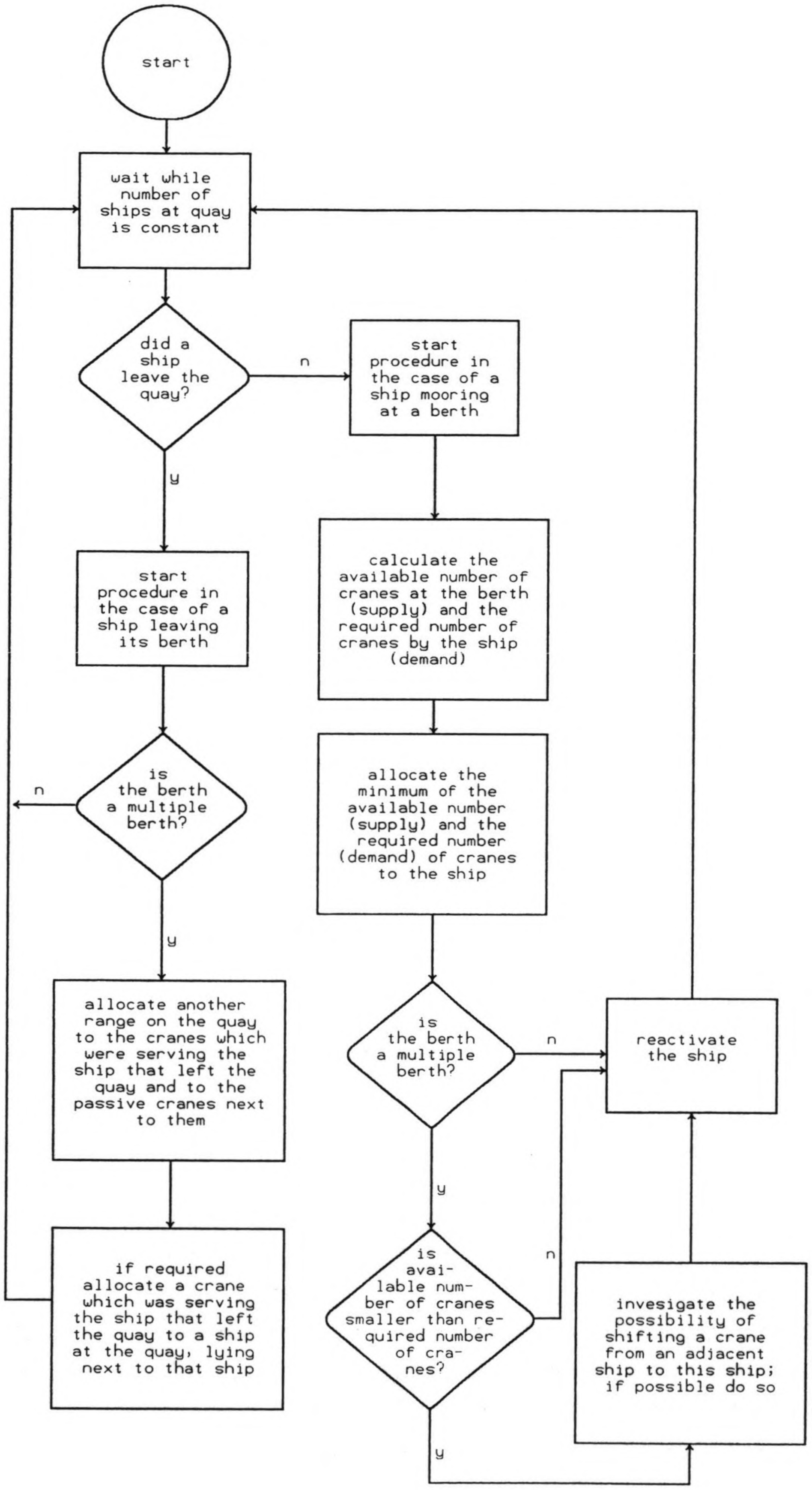
When a suitable free space has been found for the ship, which is being checked by the harbour-master, some of the ships at the berth may have been shifted and some of the cranes at the berth may have been re-allocated to facilitate this. Therefore the cranes which are not allocated to any ship at the end of the shifting of ships have to have their x-coordinates restated. This is performed in Block 5.

3.5 Terminal-master

The terminal-master has a twofold task. The first part is to allocate cranes to ships which have moored at a berth (both single and multiple berths). The second part takes place when a ship leaves its berth. The terminal-master investigates the possibility of re-allocating the cranes which were allocated to the ship, which left its (multiple) berth, to an adjacent ship at that multiple berth. As shown in the flow-chart in Figure A3.5, the terminal-master is activated at the moment the number of ships in the queue *Quay* changes (this is the queue of all ships moored at a berth and ready to be or being unloaded/loaded).

The parameters on which the terminal-master (re-)allocates cranes are the availability of cranes at the berth, the demand for cranes by the ship and the discrepancy between these two. For single berths, the available number of cranes is equal to the total number of cranes at that berth. For multiple berths, the available number of cranes to serve a ship is equal to the number of passive cranes in between the active cranes which are serving adjacent ships. The part of the quay which cranes can physically reach for serving a ship (the *range* of the crane) is defined by two x-coordinates: one coordinate for the lower boundary of the range and one

Figure A3.5: extended process of the terminal-master


start
wait while number of ships at quay is constant
did a ship leave the quay?
n
y
start procedure in the case of a ship mooring at a berth
calculate the available number of cranes at the berth (supply) and the required number of cranes by the ship (demand)
allocate the minimum of the available number (supply) and the required number (demand) of cranes to the ship
is the berth a multiple berth?
n
y
reactivate the ship
is available number of cranes smaller than required number of cranes?
n
y
invesigate the possibility of shifting a crane from an adjacent ship to this ship; if possible do so
start procedure in the case of a ship leaving its berth
is the berth a multiple berth?
n
y
allocate another range on the quay to the cranes which were serving the ship that left the quay and to the passive cranes next to them
if required allocate a crane which was serving the ship that left the quay to a ship at the quay, lying next to that ship

for the upper boundary. Therefore a crane can be allocated to serve a ship if:

$(x_{crane,min} \leq x_{ship,bow}) \wedge (x_{crane,max} \geq x_{ship,stern})$

The number of available cranes is called the *supply* of cranes. If a crane is having a breakdown it will also be allocated to a ship but will only become active after the breakdown period.

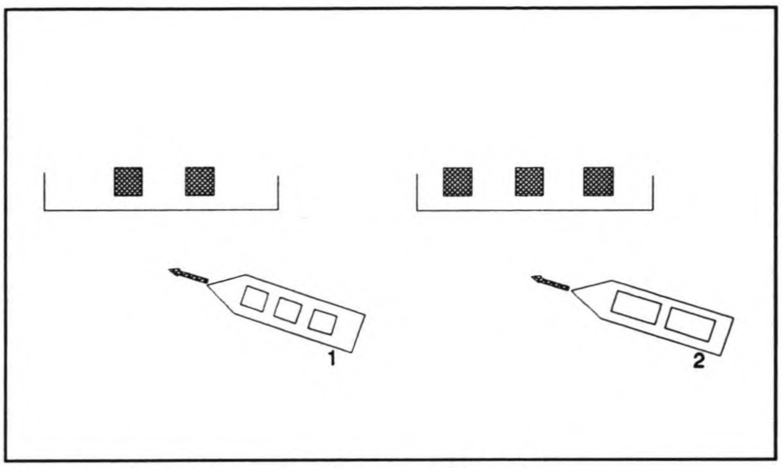


Figure A3.6: example of crane allocation

The *demand* for cranes by a ship depends on the shipclass to which the ship belongs. For each shipclass, the user has to define the maximum number of cranes that are required to serve the ship (for example the number of hatches of a ship).

When a ship joins the queue *Quay* the terminal-master checks if the berth which the ship has been sent to by the harbour-master is a single or multiple berth. If it is a single berth, the terminal-master allocates N cranes, with N being the minimum of the supply of cranes and the demand for cranes. This implies that, if more or as many cranes are required by the ship than are available at the berth (demand ≥ supply), the available number of cranes (supply) is allocated; in this case all cranes at the berth are allocated. It also implies that, if more cranes are available at the berth than are required by the ship (supply > demand), the required number of cranes (demand) is allocated; in this case, the cranes are allocated in order of the largest sum of unloading-capacity and loading-capacity. Figure A3.6 shows two examples of crane-allocation at a single berth. Two cranes are allocated to ship 1 because two cranes are available at the berth, though three are demanded by the ship (it has three hatches). Ship 2 has two hatches and requires two cranes; both demanded cranes are allocated because even more cranes (three) are available at the berth.

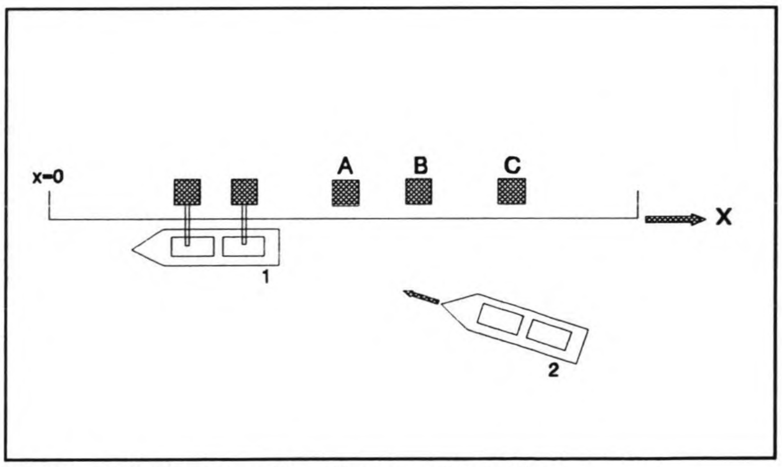


Figure A3.7: example of crane allocation

If the berth is a multiple berth, the terminal-master also allocates N cranes (with N having the same meaning and implications as in the case of the single berth); the cranes are allocated in order of physical appearance, starting with the one nearest to the bow of the ship. In the case of a multiple berth the allocation-procedure has some extra rules. If the demand is bigger than the supply, the possibility is investigated to shift a crane from an adjacent ship to the ship which is being checked by the terminal-master. If the supply is bigger than the demand, the ranges of the available cranes, which are not allocated, have to be adapted. The latter case is illustrated by an example in Figure A3.7 (before crane-allocation) and Figure A3.8 (after crane-allocation). In the example, the demand for cranes by ship 2 is two and the available supply of cranes at the quay is three. This implies that cranes A and B are allocated to ship 2 and that crane C is given a new range, in which in can possibly serve a third ship.

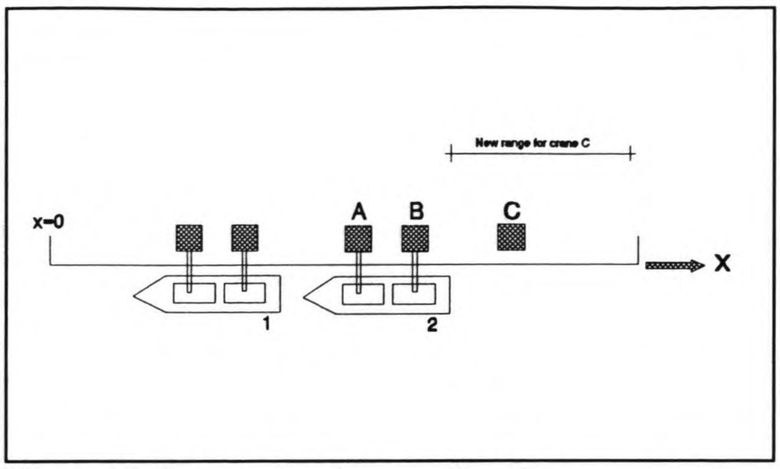


Figure A3.8: example of crane allocation

To shift a crane from an adjacent ship to the ship in question, certain conditions have to be met. If the adjacent ship has only one crane, if the discrepancy between demand and supply of the adjacent ship is bigger than the discrepancy between demand and supply of the ship in question, if the consignment-size of the adjacent ship is bigger than the consignment-size of the ship in question or if the adjacent ship is nearly completely finished with being unloaded and loaded, a crane will not be shifted. The adjacent ships on both sides are checked; the model allows one crane to be shifted. An example of shifting a crane, when a ship moors at the quay, is given in Figure A3.9. Ship 2 requires three cranes while only one is available. When ship 2 has moored, one crane will therefore be shifted from ship 1 to ship 2. After having allocated cranes to a ship, the terminal-master reactivates the ship and repeats its process from the start.

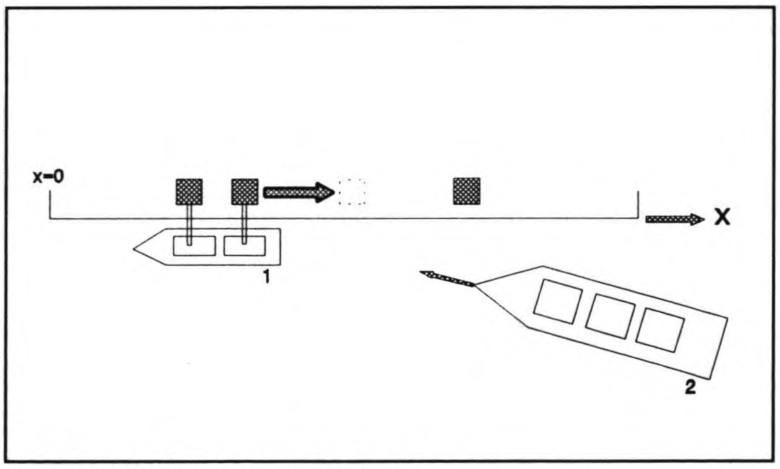


Figure A3.9: example of shifting a crane

When a ship leaves the queue *Quay*, the terminal-master must fulfil two procedures for multiple berths. The terminal-master adapts the ranges of the cranes which were serving the ship which has left the quay and the ranges of the cranes that are at rest but are standing next to the formerly active cranes. Secondly, the terminal-master investigates the possibility to shift a crane which was serving the departing ship to an adjacent ship. Again, certain conditions have to be met to shift a crane. These conditions are: the adjacent ship must have a shortage of cranes and it must not be nearly finished with unloading and loading. The adjacent ships on both sides are checked; the model allows one crane to be shifted. An example of shifting a crane, when a ship leaves the quay, is given in Figure A3.10. Ship 3 has just left its berthing-place and leaves one crane out of work. This crane is then shifted to ship 1.

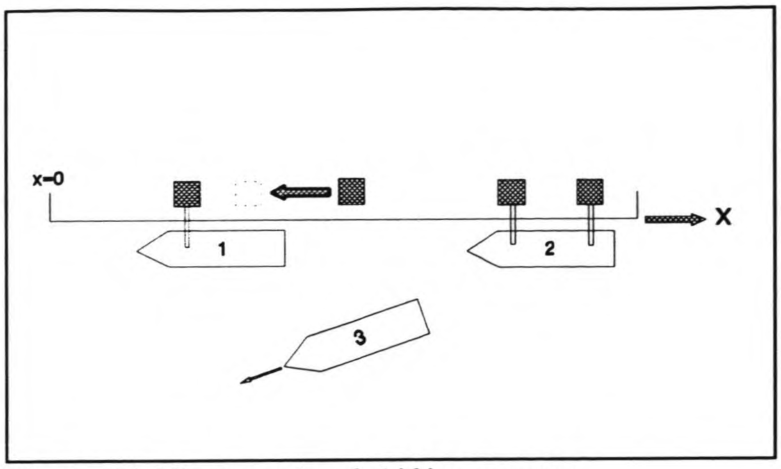


Figure A3.10: example of shifting a crane

After finishing the procedures of adapting the crane-ranges and possibly of shifting the cranes, the terminal-master repeats its process from the start.

3.5.1 Module Terminal

The task of the terminal-master is to allocate cranes to a ship. When a ship enters the port and moors at a berth, cranes are allocated to the ship; when a ship leaves its berth, cranes become available to be re-allocated. The procedures of allocation and re-allocation depend on the available number of cranes at a berth (*supply*) and of the required number of cranes for a ship (*demand*). Secondly, they depend on whether a berth is a single berth or a multiple berth. The structure of this module is based on these aspects. Lines 10-177 deal with allocation, lines 181-235 (Blocks 10-12) deal with re-allocation. The latter part can only concern multiple berths; the first part is divided in allocation of cranes at single berths (Block 2, lines 17-37) and allocation of cranes at multiple berths (Blocks 3-9, lines 38-177); the latter, in its turn, divided in:

* Supply > demand (Blocks 7-9, lines 105-177); the demanded number cranes is allocated
* Supply ≤ demand (Block 4, lines 43-60); the available number of cranes is allocated; if supply < demand the possibility of shifting a crane from an adjacent ship is checked

(Blocks 5 and 6, lines 61-104)

The purpose of this module consists of:

* Calculating values for the rates at which a ship is unloaded and loaded (= sum of the unload-capacity and the load-capacity of the allocated cranes)
* Activating the cranes which are allocated
* Shifting cranes
* Calculating values for the ranges of cranes at multiple berths. A range is defined as the part of the quay which the crane can physically reach; it is determined by two coordinates (x_{min} and x_{max}; the x-axis is parallel to the quay, with $x=0$ at one end of the quay, see also Paragraph 3.2 of this annex). Each time a crane is (re-)allocated, its range changes; in this case also the ranges of adjacent cranes must be adapted.

BLOCK 1, MODULE TERMINAL, LINES 1-16

Block 1 is the starting point of this module. In this block the terminal-master becomes active (lines 6-8). The terminal-master is activated when the length of queue *Quay* changes (line 7). When the length decreases, the departure of a ship from its berth is implied and the procedure of re-allocation is started; when the length increases, the arrival of a ship at its berth is implied and the procedure of allocation is started.

When the terminal-master is activated, it is usually to serve only one ship. However, when, after a typhoon, several ships join the queue *Quay* at once, they all need cranes. Therefore in line 11 of Block 2 a *For each..*-loop is started in which for each ship that does not have cranes allocated yet (indicated by the logical attribute *Ship_crane_alloc* having the value *False*), the allocation-procedure is performed. This loop end in line 178.

BLOCK 2, MODULE TERMINAL, LINES 17-37

In Block 2 cranes are allocated to a ship which is moored at a single berth. In the *While..End*-loop (lines 21-34) N cranes are allocated to the ship and activated, in which N stands for the minimum of the demand for cranes of the ship and the supply of cranes at the berth. The cranes are allocated in order of a decreasing sum of unload-capacity and load-capacity. For each crane the unloading-rate and the loading-rate of the ship are increased by respectively the unload-capacity and the load-capacity of the crane (lines 23 and 24). For containers, the unit of the crane-capacities is boxes/hour, so the capacities are multiplied by a factor to account for 2-TEU containers. The default-value of that factor is 1.

If a crane is suffering a breakdown, it can be allocated to a ship in advance (lines 28-31) but the capacities of that crane are not yet added to the loading-rate and the unloading-rate of the ship (this is performed in the module *Crane*). In case of a breakdown, the attribute *Crane_prev_ship* has to be given a value referring to the previous ship it was handling, because after a breakdown, certain calculations are performed in the module *Crane* for that previous ship. After finishing the *While..End*-loop, the ship is reactivated and the terminal-master repeats its process from the start.

BLOCK 3, MODULE TERMINAL, LINES 38-42

In Block 3 the number of cranes which are available to be allocated to a ship (= crane-supply) at a multiple berth is calculated. The condition that a crane must fulfil is that the range of a crane must cover the space at the berth which is occupied by the ship. This condition can be translated into:

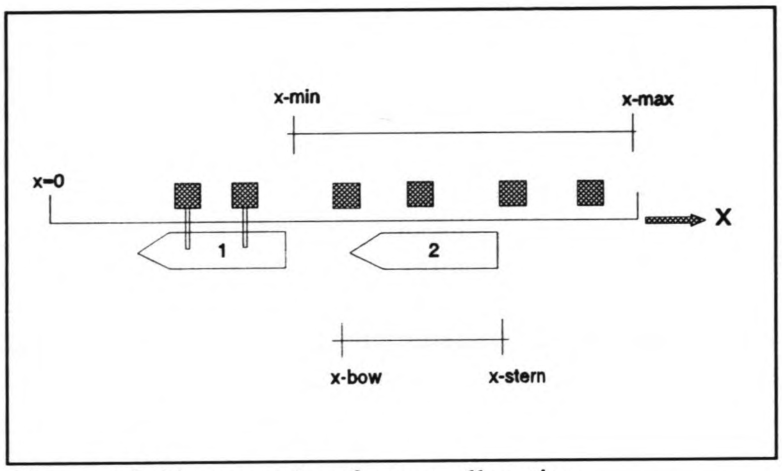


Figure A3.11: example of crane allocation

$(x_{crane,min} \leq x_{ship,bow}) \wedge (x_{crane,max} \geq x_{ship,stern})$

Figure A3.11 illustrates this condition graphically.

BLOCK 4, MODULE TERMINAL, LINES 43-60

Now that the crane-demand (a user-defined value) and the crane-supply (calculated in the previous block) are known, the cranes can be allocated. In Block 4 this is performed for the situation that the supply is smaller than or equal to the demand; in this case all available cranes are allocated and activated. The procedure for doing so is identical to the procedure of Block 2; with the addition that for ships at a multiple berth a value much be given to the attribute *Rightcrane* (line 59).

BLOCK 5, MODULE TERMINAL, LINES 61-83

The easiest situation for the terminal-master in Block 4 is that the supply and demand are equal. However, if the supply is smaller, the possibility is investigated to shift a crane from an adjacent ship to the ship in question (= the ship which is being serviced by the terminal-master). In Block 5 a check is performed to see if there are suitable adjacent ships. *Tm_ship_a* is a ship on the side with lower x-coordinates (line 68-75), *Tm_ship_b* is a ship on the side with higher x-coordinates (line 76-83). The condition for being a suitable ship have been chosen as follows:

* The discrepancy between demand and supply of the adjacent ship must be smaller than that of the ship in question
* The adjacent ship must have more than one crane allocated for unloading/loading
* The adjacent ship must have less than 75% of the loading finished (if the adjacent ship is nearly finished with unloading and loading, the crane will be shifted anyway, by means of re-allocation, when the adjacent ship leaves the berth)
* The ship in question must not have its own ships' gear

If suitable adjacent ships exist, the attributes *Tm_load_a* and *Tm_load_b* are given the values of the sum of the quantities of import- and export-cargo of respectively *Tm_ship_a* and *Tm_ship_b*; if they do not exist, *Tm_ship_a*, *Tm_ship_b*, *Tm_load_a* and *Tm_load_b* are given default values ('none' for the first two, 1e10 for the latter two).

BLOCK 6, MODULE TERMINAL, LINES 84-104

In lines 84-87 of Block 6, the ship with smallest sum of import and export of the adjacent ships is picked (called *Tm_ship_c*). If one of the adjacent ships does not exist, the other becomes *Tm_ship_c*, due to default values given in Block 5. If the ship in question has no adjacent ships, *Tm_ship_c* is 'none' and the rest of the block is skipped. If *Tm_ship_c* does exist, a crane is shifted from *Tm_ship_c* to the ship in question (lines 88-102). This is performed by changing the unloading-rate and the loading-rate of the two ships. If *Tm_ship_c* is being unloaded, the attributes *Ship_rate* (which is the attribute in the continuous function *Ship_load* and at that time has the value of *Ship_unloading_rate*) and *Ship_loading_rate* of *Tm_ship_c* need to be changed; if *Tm_ship_c* is being loaded only *Ship_rate* (which at that time has the value of *Ship_loading_rate*) needs to be changed (lines 91-93).

BLOCK 7, MODULE TERMINAL, LINES 105-141

The procedure for allocating cranes, in case of a larger supply of cranes at the berth than a demand for cranes by the ship in question, commences in Block 7. This procedure is more complicated than the allocation-procedures above, because it is not possible to select the cranes arbitrarily. The selected cranes must be adjacent and they must be in a relatively correct position on the quay compared to the position of the ship alongside the quay. A "relatively correct position" is necessary in order not to exclude cranes from serving other arriving ships, by pushing them in remote positions on the quay, with small ranges. Therefore the procedure is as follows: the first step is to select a crane (*Rightcrane*) with a sufficiently low x-coordinate but in a relatively correct position on the quay (Block 7), the second step is to allocate the demanded number of cranes, starting with *Rightcrane* and continuing with cranes with higher x-coordinates (Block 8).

The selection of *Rightcrane* is easy if the ship in question has an adjacent ship or if it is moored at either end of the quay; with other configurations the selection is more difficult. *Tm_ship_a* and *Tm_ship_b* are the possible adjacent ships, respectively with lower and with higher x-coordinates (lines 107 and 108). The different configurations of ship and cranes will now be explained:

* If the ship in question is moored at the lower end of the quay, *Rightcrane* is the first crane of the quay at that end (lines 109-111)
* If the ship in question is moored at the upper end of the quay (e.g. there is not enough space between the stern of the ship an the end of the quay for another ship), *Rightcrane* is determined by counting backwards from the last crane of the berth (lines 131-133 if the ship has an adjacent ship, lines 119-121 if the ship has no adjacent ships)
* If the ship in question has an adjacent ship with lower x-coordinates, *Rightcrane* is the first crane next to the cranes which are allocated to the adjacent ship (lines 136-140).
* If the ship in question has an adjacent ship with higher x-coordinates, *Rightcrane* is determined by counting backwards from the first crane which is allocated to that adjacent ship (lines 114-118 if the ship has one adjacent ship, lines 126-130 if the ship has two adjacent ships, of which the lower one only uses its own ships' gear).
* If the ship in question has no adjacent ships or if it only has adjacent ships which use their own ships' gear and do not have any cranes allocated, the above possibilities do not apply; in this case *Rightcrane* is determined by the *Ceil*-value of the number of cranes at the berth multiplied by the ratio of the x-coordinate of the ships' bow and the total length of the multiple berth (lines 113 and 125).

BLOCK 8, MODULE TERMINAL, LINES 142-180

Now that *Rightcrane* is selected, the cranes can be allocated. In Block 8 the allocation is performed, starting with *Rightcrane* and continuing with the adjacent cranes, on the side with higher x-coordinates. The procedure for allocation and activation of cranes (lines 143-159) is identical to the procedure of Block 2.

More cranes were available for the ship in question than were required by the ship in question. This implicates that one or more available cranes will not be allocated. The range of these non-allocated cranes must be changed; the cranes on the side of lower x-coordinates than the allocated cranes get a new upper boundary (x_{max}) and the cranes on the side of higher x-coordinates than the allocated cranes get a new lower boundary (x_{min}). This is performed in lines 160-165.

During test-runs of the program, in exceptional occasions in the procedure described in Block 7, a suitable crane (*Rightcrane*) was not selected due to an irregularity of the configuration of ships and cranes. When *Rightcrane* is not found and a ship does not have its own ships' gear, the unloading-rate and loading-rate of the crane remain zero, causing the ship to stay in the queue *Quay* and thereby disrupting the simulation. To avoid this, lines 169-174 are introduced; in the rare case that *Rightcrane* is not found for a ship, the ships' unloading-rate and loading-rate are given the value of respectively the unloading-rate and loading-rate of the first crane of the berth. This routine has a small but negligible influence on the results of the simulation. In line 177 the ship which has had cranes allocated is reactivated. In line 178 the *For each..*-loop which started in line 11 is ended. In line 179 the terminal-master repeats its process from the start.

BLOCK 9, MODULE TERMINAL, LINES 181-195

When a ship departs from a multiple berth, the range of the cranes, which were allocated to that ship, and of the adjacent cranes, which are at rest, need to be adapted. This is performed in lines 188-195 of Block 9.

BLOCK 10, MODULE TERMINAL, LINES 196-213

In Block 10 the possibility is investigated of re-allocating a crane which was allocated to the ship, that has just left a multiple berth, to an adjacent ship on the side of lower x-coordinates. If such a crane does not exist (for instance if the ship only used its own ships' gear) this block is skipped (line 200). If the adjacent ship (*Tm_ship_a*) exists, it has to submit to several conditions (line 203):

* *Tm_ship_a* must no be finished with unloading/loading
* The demand of *Tm_ship_a* for cranes must be smaller than the allocated number of cranes at that moment
* *Tm_ship_a* must have less than 75% of the unloading and loading finished (to avoid a crane being re-allocated when *Tm_ship_a* is nearly finished with unloading and loading)

If *Tm_ship_a* fulfils all conditions one crane is re-allocated to *Tm_ship_a* (lines 204-211). This is performed by changing the unloading-rate and the loading-rate of *Tm_ship_a*. If *Tm_ship_a* is being unloaded, the attributes *Ship_rate* (which is the attribute in the continuous function *Ship_load* and at that time has the value of *Ship_unloading_rate*) and *Ship_loading_rate* of *Tm_ship_a* need to be changed; if *Tm_ship_a* is being loaded only *Ship_rate* (which at that time has the value of *Ship_loading_rate*) needs to be changed.

BLOCK 11, MODULE TERMINAL, LINES 214-235

In Block 11 the possibility is investigated of re-allocating a crane which was allocated to the ship, that has just left a multiple berth, to an adjacent ship on the side of higher x-coordinates. The procedure is identical to the procedure described in the previous block. If such a crane does not exist (for instance if the ship only used its own ships' gear or if the ship had only one crane, which has been re-allocated to the adjacent ship on the other side) this block is skipped (line 218). If the adjacent ship (*Tm_ship_b*) exists, it has to submit to several conditions (line 221):

* *Tm_ship_b* must no be finished with unloading/loading
* The demand of *Tm_ship_b* for cranes must be smaller than the allocated number of cranes at that moment
* *Tm_ship_b* must have less than 75% of the unloading and loading finished (to avoid a crane being re-allocated when *Tm_ship_b* is nearly finished with unloading and loading)

If *Tm_ship_b* fulfils all conditions one crane is re-allocated to *Tm_ship_b* (lines 222-229). This is performed by changing the unloading-rate and the loading-rate of *Tm_ship_b*. If *Tm_ship_b* is being unloaded, the attributes *Ship_rate* (which is the attribute in the continuous function *Ship_load* and at that time has the value of *Ship_unloading_rate*) and *Ship_loading_rate* of *Tm_ship_b* need to be changed; if *Tm_ship_b* is being loaded only *Ship_rate* (which at that time has the value of *Ship_loading_rate*) needs to be changed.

3.6 Ship

In the port simulation model the sailing of a ship from the waiting-row at the anchorage outside the port through the entrance channel to its berth can be obstructed by two restrictions: bad weather conditions and a tidal window. A third restriction is now introduced: congestion of the entrance channel. This is performed by creating a queue called *Channel*. Ships join this queue at the moment they leave the queue *Row* and they stay in this queue for 0.25 hours, equal to the safety interval. The maximum capacity of *Channel* is set at one, implying that a ship must wait when entering *Channel* if it is already occupied. When a ship leaves *Channel* it must complete the remainder of the mooring procedure. A period of 0.25 hours is then subtracted from this procedure because it has already been spent in the queue *Channel*. If required, the same procedure can be applied when a ship leaves the port. For a two lane channel (with two way traffic) another queue can be introduced. For one way traffic the same queue can be used, with different safety intervals, depending on whether vessels are entering or leaving the port.

Annex 4 Details of the applied verification- and validation methods

4.1 Checking of modelling procedures

This paragraph contains calculations and other detailed information of the verification-checks which are described in Paragraph 6.2. It concerns the checks of continuity of cargo flows, the comparison of performances of berths and shipclasses and the requirements for storage area and storage volume. All checks are performed using output which is produced by the TEST model.

The general characteristics of the four terminals in the TEST-port are as follows:

* Terminal 1:	Containers; three identical single berths, each with two cranes; attended by container ships
* Terminal 2:	Breakbulk; two single berths, with different berthlengths; attended by general cargo ships
* Terminal 3:	Vegetable oil; one single berth with two pumps; attended by liquid bulk carriers
* Terminal 4;	Wheat; two single berths with different water-depths; attended by dry bulk carriers

4.1.1 Continuity of cargo-flows

The continuity of cargo flows has been checked for TEST 0 for a certain period of time. This period is from t=110 days to t=500 days. These moments of time have been chosen arbitrarily. Table A4.1 shows the details of the calculation. At both moments of time the storage-level and the cargo-flows have been measured. The storage-level consists of five categories:

* *Exp_open*: export cargo, open storage
* *Exp_cov*: export cargo, covered storage (only breakbulk)
* *Imp_open*: import cargo, open storage
* *Imp_cov*: import cargo, covered storage (only breakbulk)
* *Feeder*: feeder-cargo, this is cargo which enters and leaves the terminal both by ships, not by inland transport; its registration is performed separately (only containers)

Table A4.1

RESULTS OF CHECK OF CARGO CONTINUITY (EXTENDED)				
	CONTAINER [TEU]	BREAKBULK [tons]	VEGET_OIL [tons]	WHEAT [tons]
t=110				
exp_cov	0	2221	0	0
exp_open	3132	6945	22667	9200
imp_cov	0	3920	0	0
imp_open	2069	12949	33415	54599
feeder	62	0	0	0
storage total	5263	26035	56082	63799
export out	5037	30608	38162	70320
import in	4412	32557	35366	64103
export in	4817	12029	6412	16671
import out	3892	17403	33894	48238
dir. export in	0	7659	0	0
dir. import out	0	8131	0	0
t=500				
exp_cov	0	3904	0	0
exp_open	2054	13179	59919	30466
imp_cov	0	2902	0	0
imp_open	1178	9134	8633	56632
feeder	83	0	0	0
storage total	3315	29119	68552	87098
export out	163797	1003530	931966	1852550
import in	163590	1016870	903557	1816460
export in	154561	798287	937467	1820160
import out	156002	809090	926867	1798560
dir. export in	0	250878	0	0
dir. import out	0	254223	0	0
d(imp. in)	159178	984313	868191	1752357
d(exp. in)	149744	786258	931055	1803489
d(dir. exp. in)	0	243219	0	0
total in [A]	308922	2013790	1799246	3555846
d(exp. out)	158760	972922	893804	1782230
d(imp. out)	152110	791687	892973	1750322
d(dir. imp. out)	0	246092	0	0
total out	310870	2010701	1786777	3532552
d(storage)	-1948	3084	12470	23299
tot.out + d(sto) [B]	308922	2013785	1799247	3555851
A - B	0	-5	1	5

There are six cargo-flows:

* Export in: export cargo arriving by inland transport
* Import in: import cargo arriving by ship (including feeder containers)
* Export out: export cargo departing by inland transport
* Import out: import cargo departing by ship (including feeder containers)
* Direct export in: export cargo arriving by inland transport without intermediate storage on the terminal
* Direct import out: import cargo departing by inland transport without intermediate storage on the terminal

A cargo-flow is calculated by subtracting the total throughput level at t=110 from the total throughput level at t=500. In the table, the addition of the three in-coming cargo flows results in sum [A] for each cargo-type; the sum of the three out-going cargo flows and the increase of the storage level account for value [B]. The check is judged by subtracting [B] from [A].

4.1.2 Calculation of performances of berths and shipclasses

For TEST 0, TEST 1, TEST 4, TEST 5 and TEST 6, the performances of berths and shipclasses, as shown in the Report-file, have been compared. The following specific performance-values have been checked:

* TEST 0: Total occupation time of berths at each of the terminals
* TEST 1: Occupation times of berths during which there is only partial operation at the berths, due to breakdowns of cranes
* TEST 4: Occupation time of berths during which there is no operation at the berths, due to strikes
* TEST 5: Occupation time of berths during which there is no operation at the berths, due to bad weather conditions
* TEST 6: Periods of time at which berths at a terminal are not occupied due to the fact that ships, which will moor at one of the berths, are waiting outside the port because of a closed tidal window

The calculations of the comparisons are shown in Table A4.2. The comparison is performed by calculating the value for the total amount of hours of each of the five performances, for each of the four terminals, for berths as well as for shipclasses. For berths, the total amount of hours equals the **sum** of the separate performance of each of the berths at the terminal (terminal 1 has 3 berths; terminal 2 has 2 berths; terminal 3 has 1 berth; terminal 4 has

Table A4.2

RESULTS OF CHECK OF PERFORMANCES OF BERTHS AND SHIPCLASSES

TEST 0

[hours]	TERMINAL 1	TERMINAL 2	TERMINAL 3	TERMINAL 4
ann. nr. of ships [A]	751.03	367.25	104.39	207.28
av. time at quay [B]	15.78	14.16	46.27	30.51
A*B	11851.25	5200.26	4830.13	6324.11
occ. time berths	4884.00	4079.00	4830.00	2519.00
	4053.00	1122.00		3805.00
	2914.00			
sum	11851.00	5201.00	4830.00	6324.00

TEST 1: breakdown

[hours]	TERMINAL 1	TERMINAL 2	TERMINAL 3	TERMINAL 4
ann. nr. of ships [A]	751.03	367.25	104.39	207.28
av. breakdowntine [B]	1.06	0.14	3.33	0.89
A*B	796.09	51.42	347.62	184.48
berths in partial	377.00	44.00	348.00	118.00
operation due	224.00	9.00		65.00
to crane-breakdowns	194.00			
sum	795.00	53.00	348.00	183.00

TEST 4: strike

[hours]	TERMINAL 1	TERMINAL 2	TERMINAL 3	TERMINAL 4
ann. nr. of ships [A]	751.03	367.25	104.39	207.28
av. time of delays [B]	0.47	0.19	1.38	1.03
A*B	352.98	69.78	144.06	213.50
berths not in operation	177.00	71.00	144.00	72.00
due to delays	107.00	0.00		141.00
	71.00			
sum	355.00	71.00	144.00	213.00

TEST 5: typhoon

[hours]	TERMINAL 1	TERMINAL 2	TERMINAL 3	TERMINAL 4
ann. nr. of ships [A]	751.03	367.25	104.39	207.28
av. time of delays [B]	0.38	0.20	1.37	0.70
A*B	285.39	73.45	143.01	145.10
berths not in operation	108.00	36.00	143.00	108.00
due to delays	144.00	36.00		36.00
	36.00			
sum	288.00	72.00	143.00	144.00

TEST 6: tide

[hours]	TERMINAL 1	TERMINAL 2	TERMINAL 3	TERMINAL 4
ann. nr. of ships [A]	751.03	367.25	104.39	207.28
waiting due to tide	0.00	0.00	0.90	8.43
A*B	0.00	0.00	93.95	1747.37
berths not occupied	0.00	0.00	94.00	1028.00
due to tide	0.00	0.00		720.00
	0.00			
sum	0.00	0.00	94.00	1748.00

2 berths). For shipclasses, the total amount of hours equals the average annual number of ships **[A]**, multiplied by the mean performance value of one ship of a shipclass **[B]**.
The check is judged by comparing both calculations of the total amount of hours of the performances.

4.1.3 Calculation of storage area and storage volume

For each of the four cargo-types on the four terminals, hand-calculations have been made to determine the average occupied storage area c.q. volume. The calculations are described in this sub-paragraph.

On terminal 1, containers are transhipped and stored. The storage area for containers can be calculated as follows:

$$O_{cont} = \frac{Td}{365} * \frac{Ag}{h}$$

in which:

O_{cont}	= storage area for containers (m^2)
T	= annual throughput (TEU)
d	= mean dwelling time (days)
A	= net required storage area per groundslot (m^2/TEU)
g	= gross factor (for travelling lanes, etc.)
h	= stackheight

* T = 298741 TEU
* d = 5 days
* A = 2.44 * 6.10 m^2
* g = 2
* h = 1.5
* Result of hand-calculation: O = 81207 m^2
* Result of storestream 1OPEN101: O = 80954 m^2

On terminal 2, breakbulk is transhipped and stored. The storage area for breakbulk can be calculated as follows:

$$O_{br}=\frac{Td}{365}*\frac{g}{h\rho}$$

in which:

O_{br} = storage area for breakbulk (m^2)
T = annual throughput (tons)
d = mean dwelling time (days)
ρ = mean relative density (ton/m^3)
g = gross factor (for travelling lanes, etc.)
h = stackheight (m)

* 20% of the throughput is directly transported to and from the terminal, without intermediate storage $\rightarrow$
  T = 0.8 * 1843622 = 1474898 tons
* d = 7 days
* g = 2
* h = 2.0 m.
* ρ = 0.8 ton/m^3
* Result of the hand-calculation: O = 35358 m^2
* Result of storestream 2OPEN201 + 2COV201:
  O = 26933(open) + 9044 (covered) = 35977 m^2

For liquid bulk as well as dry bulk, the same formula can be used to calculate the average occupied storage volume:

$$V=\frac{Td}{365}*\frac{g}{\rho}$$

in which:

V = storage volume for liquid bulk/dry bulk (m^3)
T = annual throughput (tons)
d = mean dwelling time (days)
ρ = mean relative density (ton/m^3)
g = gross factor (for travelling lanes, etc.)

Calculation for vegetable oil on terminal 3:
* T = 1674915 tons
* d = 9 days

* $g = 2$
* $\rho = 0.5$ ton/m^3
* Result of hand-calculation: O = 165196 m^3
* Result of storestream 3VOL301: O = 185790 m^3

Calculation for wheat on terminal 4:
* T = 1674915 tons
* 30% of the cargo is has a mean dwelling time of 9 days and 70% has a mean dwelling time of 6.07 days $\rightarrow$ d = 6.95 days
* $g = 2$
* $\rho = 0.8$ ton/m^3
* Result of hand-calculation: O = 159350 m^3
* Result of storestream 4VOL401: O = 185770 m^3

4.2 Queuing theory

Table A4.3 shows the values of the average waiting time and average service time for terminal 1 to 4 in TEST 0 to TEST 8. It also shows the results of the queuing theory calculations. These calculations are the topic of Paragraph 4.2.1 (Terminal 3) and Paragraph 4.2.2 (Terminal 1).

The restrictions which are applied in the several runs of the TEST model are as follows:

* TEST 0: No restrictions
* TEST 1: 20 hour-breakdowns of cranes at an average interval time of 500 working hours per crane
* TEST 2: No inland transport on weekend days
* TEST 3: No shifts at the terminal during weekend days
* TEST 4: Two day-strikes at average interval times of 75 days
* TEST 5: Two day-typhoons at average intervaltimes of 75 days obstructing both landside- and marine-activities
* TEST 6: A tidal window, restricting the classes which attend terminal 3 and 4
* TEST 7: 3 six hour-shifts per day instead of 3 eight hour-shifts per day
* TEST 8: At terminal 1 a multiple berth with a capacity of three ships, replacing the three single berths

Table A4.3

RESULTS OF CHECK WITH QUEUING THEORY (EXTENDED)

	SHIPCLASS:	1	2	3	4		ROWLENGTH:
	A: Average waiting time [hours] B: Average mooring time [hours] C: Average service time [hours]						Average Bound.95% [number of ships]
TEST 0	A	2.80	6.75	37.52	7.72		1.67
	B	4.00	3.99	6.07	5.98		4.86
	C	15.78	14.16	46.27	30.51		
Queuing	A	2.71	x	38.59	x		x
theory:	B+C	19.78	x	52.34	x		x
TEST 1	A	3.22	6.77	42.15	8.75		1.78
	C	16.72	14.21	47.93	30.81		4.86
TEST 2	A	2.80	6.75	37.52	7.72		1.67
	C	15.78	14.16	46.27	30.51		4.86
TEST 3	A	22.18	21.28	131.35	35.65		5.69
	C	23.99	21.83	65.78	45.92		12.90
TEST 4	A	3.62	7.99	43.13	9.36		1.97
	C	16.25	14.36	47.65	31.54		5.63
TEST 5	A	4.40	7.05	38.94	7.92		1.91
	C	16.16	14.36	47.64	31.21		6.11
TEST 6	A	2.80	6.75	40.47	16.57		1.92
	C	15.78	14.16	46.27	30.46		4.92
TEST 7	A	34.08	30.50	291.68	52.19		9.31
	C	26.98	24.62	77.43	51.17		19.33
TEST 8	A	2.83	6.75	37.52	7.72		1.68
	C	15.99	14.16	46.27	30.51		4.86
TEST-A	A	4.00	7.73	48.15	16.79		1.73
	B	4.00	4.00	6.07	6.00		4.91
	C	15.92	14.28	46.27	30.46		
Queuing	A	3.54	x	52.06	x		x
theory:	B+C	19.92	x	52.34	x		x
TEST-A1	A	4.53	7.87	53.89	17.22		1.83
	C	16.20	14.38	48.93	30.70		4.86
TEST-A4	A	8.08	8.26	52.28	19.13		2.01
	C	16.16	14.59	48.82	31.51		5.75
TEST-A5	A	5.06	8.87	52.46	18.07		1.95
	C	16.13	14.62	48.39	31.01		5.47

4.2.1 Terminal 3

Data:
* Mean inter-arrivaltime: 3.8 days
* Mean service-time: 46.27 hours
* Mean mooring-time: 6.07 hours
* Ship-arrivals: negative exponential distribution
* Service-time depends on the consignment-size of ships. The consignment-size is drawn from a normal distribution (TEST) and from an exponential distribution (TEST-A).
* 1 berth

Calculation for TEST
* Queue-system $M/E_k/1$
* k = 10
* Arrival-rate: A = 1/3.8 = 0.263
* Service-rate: S = 24/(6.07+46.27) = 0.459
* R = A/S = 0.573
* Utilization: U = R = 0.573
* Formula for W (average waiting time):

$$W=\frac{1+k}{2k}*\frac{R}{(1-R)S}$$

* W = 1.608 days = 38.59 hours

Calculation for TEST-A:
* Queue-system $M/E_k/1$
* k = 2.068
* Arrival-rate: A = 1/3.8 = 0.263
* Service-rate: S = 0.459
* R = A/S = 0.573
* Utilization: U = R = 0.573
* W = 2.169 days = 52.06 hours

4.2.2 Terminal 1

Data:
* Mean inter-arrivaltime: 0.5 days
* Mean service-time: 15.79 hours

* Mean mooring-time: 4.00 hours
* Ship-arrivals: negative exponential distribution
* Service-time depends on the consignment-size of ships. The consignment-size is drawn from a normal distribution (TEST) and from an exponential distribution (TEST-A).
* 3 berths

Calculation for TEST:
* Queue-system $M/E_k/N$
* $k=10$ and $N=3$
* Arrival-rate: $A = 1/0.5 = 2$
* Service-rate: $S = 24/(4.00+15.79) = 1.213$
* $R = A/S = 1.649$
* Utilization: $U = R/N = 0.550$
* For a $E_l/E_k/N$-system, W_n can be approximated by linear interpolation on n_a and n_s , using the queuing systems M/M/N, D/M/N, M/D/N, D/D/N (with $n_a = 1/l$ $n_s = 1/k$). The approximation is:

$$W_n=(1-n_a)n_sW_n(0,1,u)+n_a(1-n_s)W_n(1,0,u)+n_an_sW(1,1,u)$$

in which:

$W_n(1,1,u)$ = average waiting time in M/M/n with utilisation u
$W_n(1,0,u)$ = average waiting time in M/D/n with utilisation u
$W_n(0,1,u)$ = average waiting time in D/M/n with utilisation u

* For a $M/E_k/N$-system the same approximation can be applied, using $n_a = 1$
* $n_s = 1/k = 0.1$
* $u = U = 0.550$
* $W_3(1,0.1,0.5) = 0 + 1*0.9*W_3(1,0,0.5) + 1*0.1*W_3(1,1,0.5)$
  $= 0 + 0.9*0.0872 + 0.1*0.1579 = 0.0943$
* $W_3(1,0.1,0.6) = 0 + 1*0.9*W_3(1,0,0.6) + 1*0.1*W_3(1,1,0.6)$
  $= 0 + 0.9*0.1584 + 0.1*0.2956 = 0.1722$
* $W_3(1,0,0.5)$ and $W_3(1,0,0.6)$ are found in Table A4.4;
  $W_3(1,1,0.5)$ and $W_3(1,1,0.6)$ are found in Table A4.5
* Linear interpolation of $W_3(1,0.1,0.5)$ and $W_3(1,0.1,0.6)$:
  $W_3(1,0.1,0.550) = 0.1371$
* $W = 0.1371 * 19.79 = 2.71$ hours

Calculation for TEST-A:

* Queue-system $M/E_k/N$
* $k=2$ and $N=3$
* Arrival-rate: $A = 1/0.5 = 2$
* Service-rate: $S = 24/(4.00+19.92) = 1.205$
* $R = A/S = 1.660$
* Utilization: $U = R/N = 0.553$
* For a $E_l/E_k/N$-system, W_n can be approximated by linear interpolation on n_a and n_s , using the queuing systems M/M/N, D/M/N, M/D/N, D/D/N (with $n_a = 1/l$ $n_s = 1/k$). For a $M/E_k/N$-system the same approximation can be applied, using $n_a = 1$ (see also calculation for TEST)
* $n_s = 1/k = 0.5$
* $u = U = 0.553$
* $W_3(1,0.5,0.5) = 0 + 1*0.5*W_3(1,0,0.5) + 1*0.5*W_3(1,1,0.5)$
  $= 0 + 0.5*0.0872 + 0.5*0.1579 = 0.1226$
* $W_3(1,0.5,0.6) = 0 + 1*0.5*W_3(1,0,0.6) + 1*0.5*W_3(1,1,0.6)$
  $= 0 + 0.5*0.1584 + 0.5*0.2956 = 0.2270$
* $W_3(1,0,0.5)$ and $W_3(1,0,0.6)$ are found in Table A4.4;
  $W_3(1,1,0.5)$ and $W_3(1,1,0.6)$ are found in Table A4.5
* Linear interpolation of $W_3(1,0.5,0.5)$ and $W_3(1,0.5,0.6)$:
  $W_3(1,0.5,0.553) = 0.1779$
* $W = 0.1779 * 19.92 = 3.54$ hours

Table A4.4

Average waiting time of customers in the queue M/D/n
(in units of the average service time)

utilization	Number Servicing points									
	1	2	3	4	5	6	7	8	9	10
0.1	0.055	0.0062	0.0009	0.0002	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
0.2	0.125	0.0242	0.0066	0.0021	0.0007	0.0002	0.0001	0.0000	0.0000	0.0000
0.3	0.214	0.0553	0.0201	0.0085	0.0039	0.0019	0.0009	0.0005	0.0002	0.0001
0.4	0.333	0.1033	0.0450	0.0227	0.0124	0.0072	0.0043	0.0026	0.0017	0.0011
0.5	0.500	0.1767	0.0872	0.0497	0.0307	0.0199	0.0135	0.0093	0.0066	0.0047
0.6	0.750	0.2930	0.1584	0.0984	0.0661	0.0467	0.0342	0.0257	0.0197	0.0154
0.7	1.167	0.4936	0.2862	0.1897	0.1355	0.1016	0.0788	0.0627	0.0508	0.0419
0.8	2.000	0.9030	0.5537	0.3860	0.2890	0.2265	0.1833	0.1519	0.1282	0.1098
0.9	4.500	2.0138	1.2887	0.9340	0.7237	0.5848	0.4894	0.4164	0.3606	0.3175

Table A4.5

AVERAGE WAITING OF CUSTOMERS IN THE QUEUE *M/M/n*, IN UNITS OF AVERAGE SERVICE TIME

Utilisation (*u*)	*Number of Servers (n)*								
	2	3	4	5	6	7	8	9	10
0·1	0·0101	0·0014	0·0002	0·0000	0·0000	0·0000	0·0000	0·0000	0·0000
0·2	0·0417	0·0103	0·0030	0·0010	0·0003	0·0001	0·0000	0·0000	0·0000
0·3	0·0989	0·0333	0·0132	0·0058	0·0027	0·0013	0·0006	0·0003	0·0002
0·4	0·1905	0·0784	0·0378	0·0199	0·0111	0·0064	0·0039	0·0024	0·0015
0·5	0·3333	0·1579	0·0870	0·0521	0·0330	0·0218	0·0148	0·0102	0·0072
0·6	0·5625	0·2956	0·1794	0·1181	0·0819	0·0589	0·0436	0·0330	0·0253
0·7	0·9608	0·5470	0·3572	0·2519	0·1867	0·1432	0·1128	0·0906	0·0739
0·8	1·7778	1·0787	0·7455	0·5541	0·4315	0·3471	0·2860	0·2401	0·2046
0·9	4·2632	2·7235	1·9693	1·5250	1·2335	1·0285	0·8769	0·7606	0·6687

$$\text{utilisation} = \frac{\text{Av. Service Time}}{(n \times \text{Av. Arrival Interval})}$$

n = number of servers

4.3 Campana (Argentina), Container terminal

4.3.1 Introduction

A verification of HASPORT-II has been performed using data of a container terminal in Campana, Maderera, Argentina. During a study on infrastructural and operation recommendations for that terminal, a simulation was performed using computer simulation model TTACTE. With this model storage area requirements and receipts and deliveries of containers with inland transport can be calculated. The results of the simulation are in a NEDECO-report, dated April 1992. Two simulation-runs have been performed; the first with an annual throughput of 32000 containers (run A), the second with an annual throughput of 64000 containers (run B). The results of the simulation of both models have been compared.

4.3.2 Description of the terminal

Campana is a terminal at which only containers are transhipped. One berth is available for unloading and loading. There are no restrictions on the terminal operations due to a tidal window, bad weather, strikes or absence of inland transport or shifts on weekend-days.

The storage characteristics are as follows:

Mean stackheight:

Import :	1.5
Export :	2.5
Empties:	3.5

Percentage of 40-feet containers:

Import :	15
Export :	15

Percentage of empties:

Import :	59
Export :	15

Arrival-pattern:

Day :	-9	-8	-7	-6	-5	-4	-3	-2	-1
Road(%):	0	5	10	20	30	20	10	5	0

Departure-pattern:

Day :	1	2	3	4	5	6	7	8	9
Road(%):	0	0	5	10	20	30	20	10	5

The average number of import-containers and the average number of export-containers are both 220 containers per vessel. The standard deviation of both is zero. Due to the throughput-figures for runs A and B, the mean inter-arrivaltimes are respectively 5 days and 2.5 days.

4.3.3 Description of computer simulation model TTACTE

TTACTE is a simulation model, with which, on the basis of the set of data in the previous sub-paragraph, calculations can be performed for container terminals. Storage area requirements and inland transport requirements can be determined. The inland transport modes road, rail and inland waterway can be applied; in the above example only road is used.

Ships are generated at inter-arrivaltimes, which are drawn from a Poisson-distribution. The servicetime for the unloading and loading of a ship has a standard length: it is one day for each ship. This implies that each berth (in the example: one) can handle one ship per day.

The output of TTACTE consists of the frequency distributions of the numbers of receipts and deliveries of containers and the frequency distributions of the required groundslots. The receipts and deliveries are divided into each mode of transport and the requirements for surface area are split up into stacks for import containers, export containers, empty containers and the sum of import and export containers.

4.3.4 Comparison of results

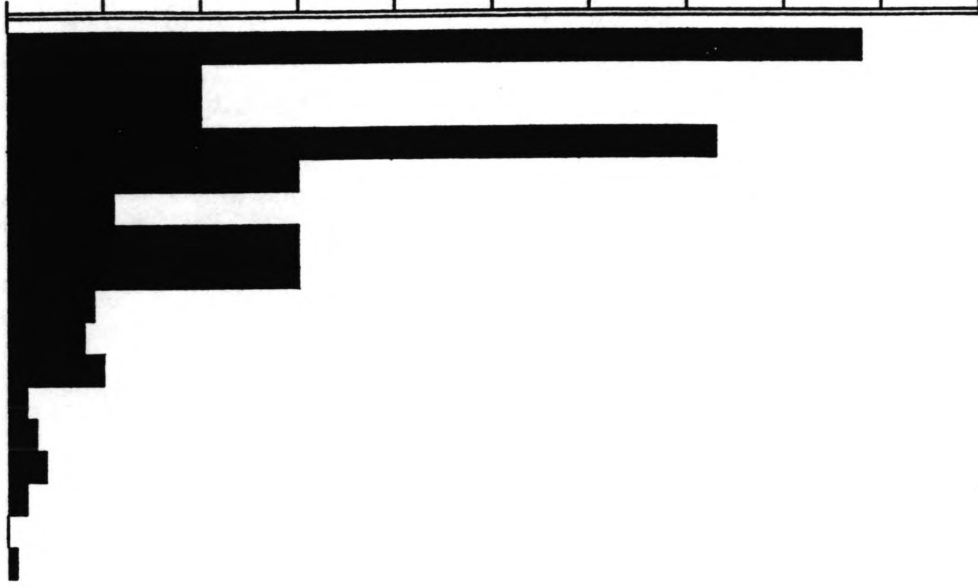
The storage-characteristics which are described above are sufficient to function as input-data for TTACTE. For HASPORT-II more input-information is required; the values for these data have been chosen in such a way that they not interfere with making a comparison between the results of the two models.

Only one aspect confuses the comparison to a small extent. When a ship leaves its berth in HASPORT-II, a second one can moor and be handled straight away. In TTACTE, a maximum of one ship can be served per day. So, when two ships arrive at a short interval (for example two ships arrive on one day), in HASPORT-II the first ship is served, directly followed by the second. In TTACTE the second has to wait until the next day to unloaded and loaded at the berth.

Figures A4.1 to A4.6 show histograms of the frequency distributions of respectively the number of occupied export-groundslots, import-groundslots and empties-groundslots for run A and the number of occupied export-groundslots, import-groundslots and empties-groundslots for run B. The top histogram in each figure shows the result of TTACTE, the bottom histogram shows the result of HASPORT-II.

The figures indicate that the frequency distributions which are calculated by both models show very similar patterns for export, import and empties. The main difference between the outcome of the two models is that the distributions which are calculated by TTACTE show higher maximum values than the distribution which are calculated by HASPORT-II. This discrepancy can be contributed to the methods of both models in which ships are handled. As explained above, TTACTE can only deal with one ship per day. In case two ships arrive on one day, the second is dealt with one day later. This implies that its cargo stays in the port-system for an extra day and is counted as lying on the stacks. Due to the fact that the stacks are already quite occupied due to the arrival of the first ship, the extreme values are created. HASPORT-II deals with this situation more realistically, causing the extreme values to be slightly decreased.

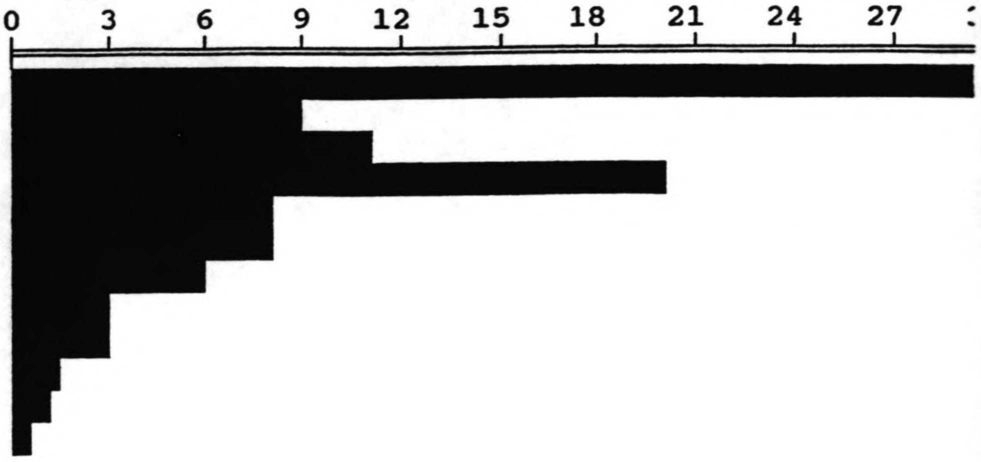
```
 PERSONAL PROSIM HISTOGRAM FACILITY == FILE STOR2_CA== SELECTION 1EXSL101    =
Licensed to Cicil Engineering DUT te Delft
================================================================================
                          Mean       :        79.193619 Minimum:          0.00000
                          Deviation:          67.522049 Maximum:        328.67999
  Upper        Cum    400 90%        :       166.000000 95%     :       206.66667
  Bound        Perc Entries 0    3    6    9   12   15   18   21   24   27   3

   20.000   26.25    105
   40.000   32.00     23
   60.000   38.00     24
   80.000   59.75     87
  100.000   68.50     35
  120.000   71.75     13
  140.000   80.50     35
  160.000   89.25     35
  180.000   91.75     10
  200.000   94.00      9
  220.000   97.00     12
  240.000   97.50      2
  260.000   98.25      3
  280.000   99.25      4
  300.000   99.75      2
  320.000   99.75      0
  340.000  100.00      1
```



```
         STORAGE REQUIREMENTS EXPORT
      0         1         2         3         4         5
CLASS I12345678901234567890123456789012345678901234567890
---------------------------------------------------------
-  20 I******************************
-  40 I******
-  60 I*******
-  80 I*******
- 100 I**********************
- 120 I*****
- 140 I*****
- 160 I*****
- 180 I********
- 200 I**
- 220 I**
- 240 I**
- 260 I*
- 280 I
- 300 I
- 320 I
- 340 I
- 360 I
- 380 I
- 400 I
```

Figure A4.1

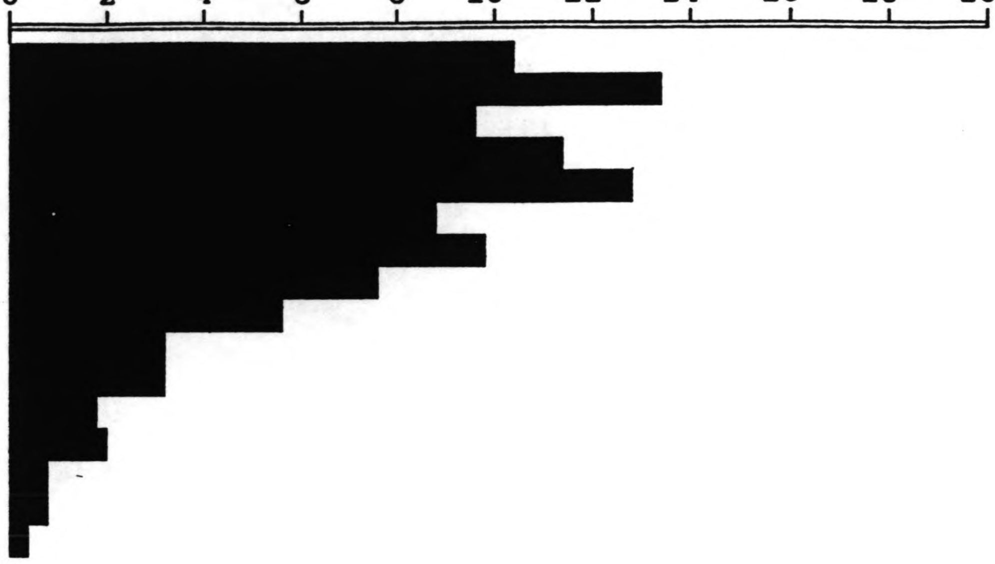
```
= PERSONAL PROSIM HISTOGRAM FACILITY = FILE STOR2_CA= SELECTION 1IMSL101
 Licensed to Cicil Engineering DUT te Delft

                                 Mean      :      60.449032 Minimum:         0.000
                                 Deviation:       53.053173 Maximum:       228.506
  Upper        Cum       400     90%       :     135.000000 95%     :      165.000
  Bound       Perc  Entries  0    3     6     9    12    15    18    21    24    27

    20.000   29.75      119
    40.000   38.50       35
    60.000   49.50       44
    80.000   69.50       80
   100.000   77.50       32
   120.000   85.50       32
   140.000   91.50       24
   160.000   94.25       11
   180.000   97.25       12
   200.000   98.50        5
   220.000   99.50        4
   240.000  100.00        2
```



```
          STORAGE REQUIREMENTS IMPORT
      0         1         2         3         4         5
CLASS I12345678901234567890123456789012345678901234567890
----------------------------------------------------------
-  20 I************************
-  40 I*****
-  60 I*********
-  80 I**********************
- 100 I*******
- 120 I******
- 140 I*************
- 160 I****
- 180 I**
- 200 I****
- 220 I**
- 240 I*
- 260 I*
- 280 I*
- 300 I
- 320 I
- 340 I
- 360 I
- 380 I
- 400 I
```

Figure A4.2

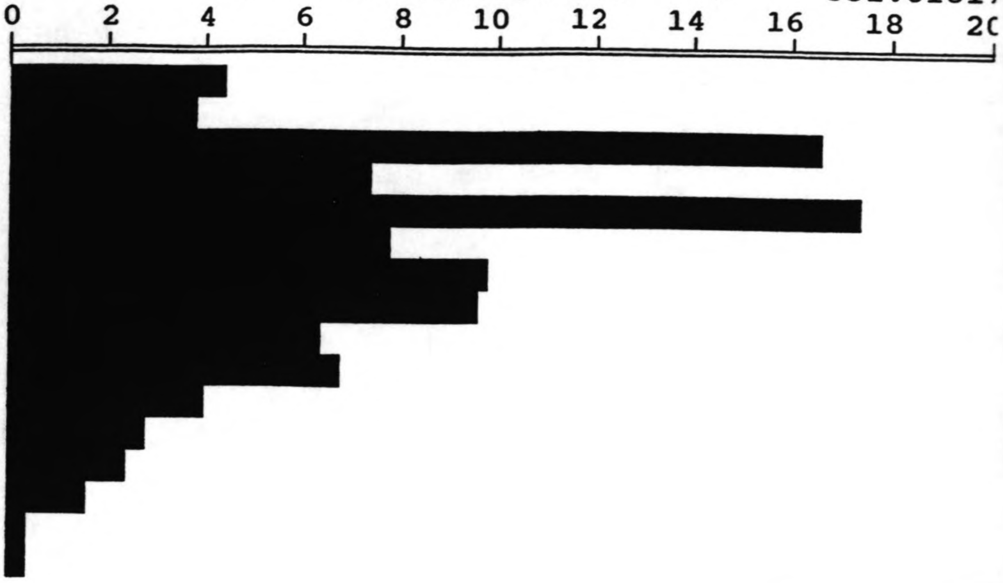
```
ERSONAL PROSIM HISTOGRAM FACILITY ══ FILE STOR2_CA══ SELECTION 1EMSL101
censed to Cicil Engineering DUT te Delft

                               Mean       :      48.866402 Minimum:           0.000000
                               Deviation:        33.049740 Maximum:         155.100006
 Upper       Cum       400     90%        :      95.000000 95%     :        112.857140
 Bound       Perc  Entries  0      2      4      6      8     10     12     14     16     18     20%

  10.000    10.25       41
  20.000    23.50       53
  30.000    33.00       38
  40.000    44.25       45
  50.000    57.00       51
  60.000    65.75       35
  70.000    75.50       39
  80.000    83.00       30
  90.000    88.50       22
 100.000    91.50       12
 110.000    94.50       12
 120.000    96.25        7
 130.000    98.25        8
 140.000    99.00        3
 150.000    99.75        3
 160.000   100.00        1
```



```
-    U
            TOTAL STORAGE REQUIREMENTS EMPTIES
        0         1         2         3         4         5
  CLASS I12345678901234567890123456789012345678901234567890
  ---------------------------------------------------------
  -  10 I********
  -  20 I*********
  -  30 I********
  -  40 I******
  -  50 I************
  -  60 I**********
  -  70 I*********
  -  80 I******
  -  90 I*******
  - 100 I**********
  - 110 I****
  - 120 I***
  - 130 I**
  - 140 I**
  - 150 I*
  - 160 I*
  - 170 I*
  - 180 I
  - 190 I
  - 200 I
 1
```

Figure A4.3

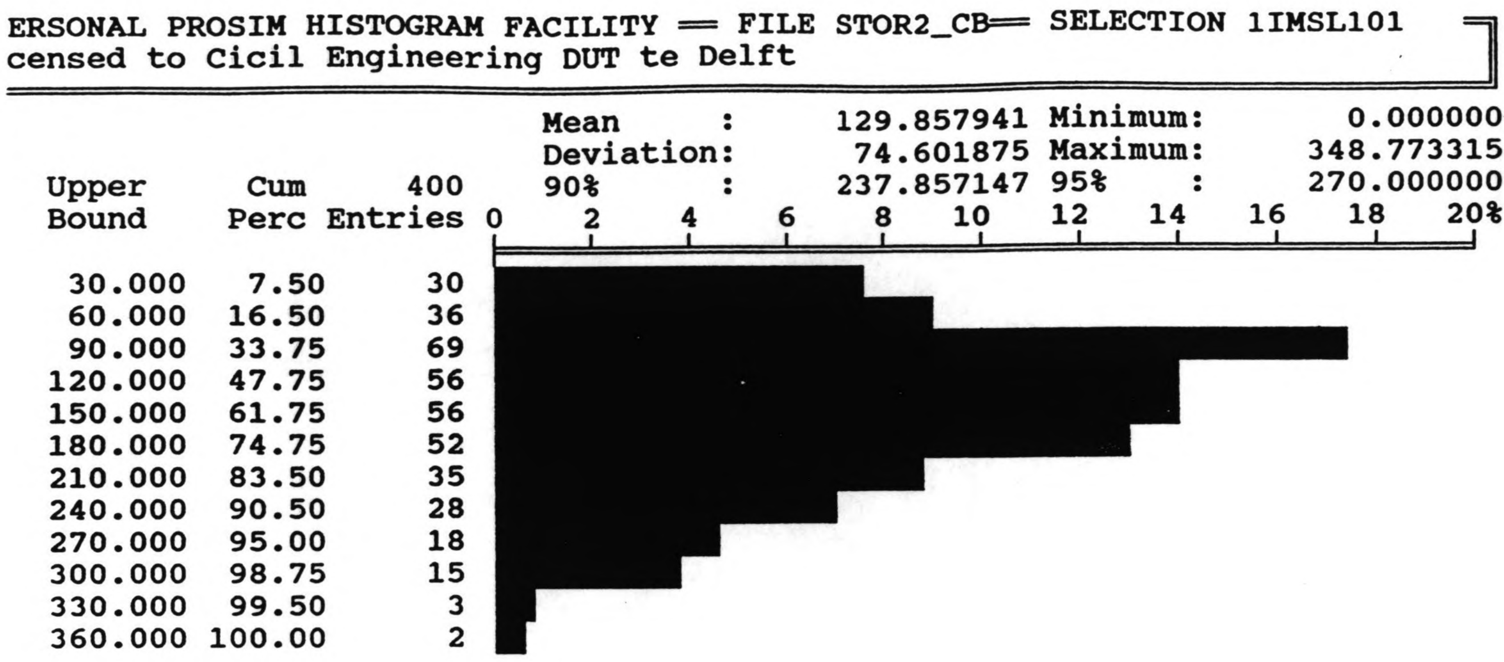
```
= PERSONAL PROSIM HISTOGRAM FACILITY ═ FILE STOR2_CB═ SELECTION 1EXSL101   =
 Licensed to Cicil Engineering DUT te Delft

                                  Mean     :       173.314789 Minimum:         0.0000
                                  Deviation:        95.122810 Maximum:       460.1520
    Upper        Cum        400   90%      :       307.500000 95%     :      351.8181
    Bound        Perc  Entries  0     2     4     6     8    10    12    14    16    18    2
```



```
     30.000     4.25       17
     60.000     8.00       15
     90.000    24.50       66
    120.000    31.75       29
    150.000    49.00       69
    180.000    56.75       31
    210.000    66.50       39
    240.000    76.00       38
    270.000    82.25       25
    300.000    89.00       27
    330.000    93.00       16
    360.000    95.75       11
    390.000    98.00        9
    420.000    99.50        6
    450.000    99.75        1
    480.000   100.00        1
```

```
0
           STORAGE REQUIREMENTS EXPORT
      0         1         2         3         4         5
CLASS I12345678901234567890123456789012345678901234567890
-----------------------------------------------------------
-  30 I********
-  60 I****
-  90 I******
- 120 I*************
- 150 I********
- 180 I**********
- 210 I***********
- 240 I*********
- 270 I*******
- 300 I*********
- 330 I***
- 360 I***
- 390 I****
- 420 I**
- 450 I*
- 480 I**
- 510 I
- 540 I*
- 570 I
- 600 I
```

Figure A4.4

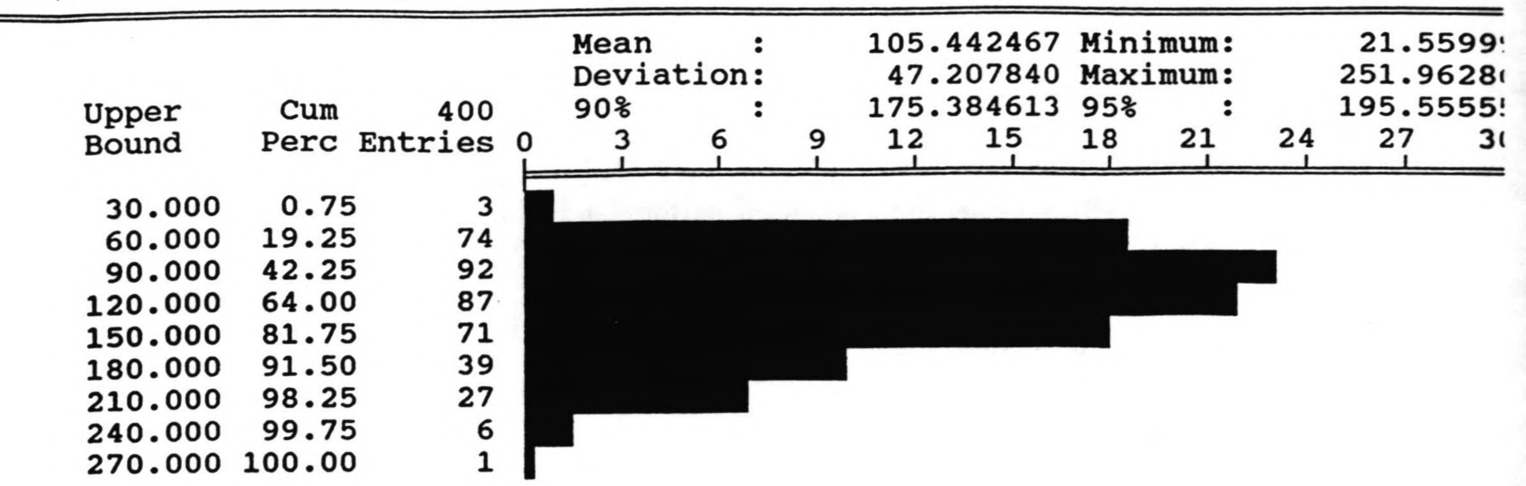


```
ERSONAL PROSIM HISTOGRAM FACILITY ══ FILE STOR2_CB══ SELECTION 1IMSL101
censed to Cicil Engineering DUT te Delft

                                  Mean      :      129.857941 Minimum:        0.000000
                                  Deviation:        74.601875 Maximum:      348.773315
 Upper       Cum       400        90%       :      237.857147 95%     :     270.000000
 Bound       Perc  Entries  0    2    4    6    8   10   12   14   16   18   20%

  30.000     7.50       30
  60.000    16.50       36
  90.000    33.75       69
 120.000    47.75       56
 150.000    61.75       56
 180.000    74.75       52
 210.000    83.50       35
 240.000    90.50       28
 270.000    95.00       18
 300.000    98.75       15
 330.000    99.50        3
 360.000   100.00        2
```

```
          STORAGE REQUIREMENTS IMPORT
         0         1         2         3         4         5
CLASS I!^345678901234567890123456789012345678901234567890
-----------------------------------------------------------
-   30 I*******
-   60 I***
-   90 I************
- 120 I*******
- 150 I********
- 180 I**************
- 210 I*********
- 240 I*************
- 270 I*******
- 300 I******
- 330 I*****
- 360 I**
- 390 I***
- 420 I**
- 450 I*
- 480 I*
- 510 I
- 540 I*
- 570 I
- 600 I
```

Figure A4.5

```
= PERSONAL PROSIM HISTOGRAM FACILITY ═ FILE STOR2_CB═ SELECTION 1EMSL101   :
Licensed to Cicil Engineering DUT te Delft
```



```
                                   Mean       :    105.442467 Minimum:       21.5599
                                   Deviation:       47.207840 Maximum:      251.9628
  Upper       Cum        400       90%        :    175.384613 95%      :     195.5555
  Bound      Perc  Entries 0      3      6      9     12     15     18     21     24     27    3

    30.000   0.75        3
    60.000  19.25       74
    90.000  42.25       92
   120.000  64.00       87
   150.000  81.75       71
   180.000  91.50       39
   210.000  98.25       27
   240.000  99.75        6
   270.000 100.00        1
```

```
        TOTAL STORAGE REQUIREMENTS EMPTIES
      0         1         2         3         4         5
CLASS I12345678901234567890123456789012345678901234567890
----------------------------------------------------------
-  30 I***
-  60 I********
-  90 I************
- 120 I*****************
- 150 I*******************
- 180 I*******************
- 210 I**************
- 240 I******
- 270 I*****
- 300 I**
- 330 I*
- 360 I*
- 390 I
- 420 I
- 450 I
- 480 I
- 510 I
- 540 I
- 570 I
- 600 I
1
```

Figure A4.6

4.4 Rotterdam-Waalhaven, Terminal Pier 2

4.4.1 Introduction

For HASPORT-II, a verification has been performed through a simulation of Terminal Waalhaven-Pier 2 in the port of Rotterdam. In may 1990 a lay-out study was performed for a extension of the terminal. Based on the report of this study, a set of input-data was created for the model. After performing a simulation-run the output of the run was compared with information from the report.

Some of the data which are required for input in HASPORT-II were not mentioned directly in the report. In this case, the information was found by making realistic assumptions or by transforming other information from the report into the required data. In sporadic cases this led to not so realistic values (for example: consignment size), but these have retained in order to facilitate the comparison.

4.4.2 Description of the terminal

The terminal is used for transport and storage of general cargo goods, wood, iron and steel, non-ferro goods, RoRo-trucks and containers. The transhipment of most of these goods is done indirectly, only part of the iron/steel and general cargo is transported directly to and from barges and wagons. Most goods are transported to and from the terminal on trucks and barges, only a small part is by rail. The terminal has a quay of 1000 meters and facilities for open as well as covered storage.

The simulation-run has been made using the forecasted throughput-figures for 1992. These figures are as follows:

General cargo	315000 tons
Wood	240000 tons
Iron/steel	225000 tons
Non-ferro	10000 tons
Containers	220000 tons
Roro	50000 tons
TOTAL	1060000 tons

4.4.3 Model input

Waalhaven-Pier 2 is modelled as one terminal; the restrictions on the port operations (tidal window, breakdowns, etc.) are switched off during of the simulation. On both weekend-days inland transport is available. The simulation-time is three years (1100 days), which is approximately equal to the arrival of 1500 ships. The cargo-types are divided into four categories: general cargo, wood, iron/steel/non-ferro, containers/roro. This division is similar to the one made in the HASKONING-report; it will facilitate the comparison between the study-results and HASPORT-II-results.

The total available quaylength is 1000 m; the average length (at berth) of the ships is 140 m. This means that a maximum of seven ships can moor at the berth. One berth at the terminal is used by the container-ships and roro-ships. This is modelled as a single berth. The rest of the quay is modelled a multiple berth. This multiple berth cannot have a capacity of six ships because the maximum number of ships at a multiple berth in HASPORT-II is four. These two berths are sufficient, regarding the inter-arrivaltimes of ships (see below).

The percentages of the total quantity of goods which are stored in sheds/warehouses are 59 % for general cargo, 15% for wood and 45% for iron/steel/non-ferro.

In the report calculations for the storage requirements are made using a so-called "package-factor". This represents the gross amount of area in square meters needed for one ton of cargo. In HASPORT-II, for breakbulk three parameters are required to simulate the influence of the package-factor: gross factor, relative density, mean stackheight. The values of these three factors have been chosen in such a way that their influence is identical to that of the package-factor.

The report uses "tons" as a unit for the containers. HASPORT-II uses TEU's so the input-information for containers has to be changed into different units. The average package-factor for containers (220000 tons; 0.5 m^2/ton) and roro-goods (50000 tons; 1.5 m^2/ton) is 0.69 m^2/ton. With a mean stackheight (import and export) of 1.5 containers, a slot-area of (2.44*6.10=) 14.88 m^2/TEU and an average weight per container of 20 tons the gross factor can be calculated to be 1.39. The percentage of forty feet-containers is zero, the percentage of empty container as well (the report has no information on quantities and dwell times of empties).

The report does not give a clear opinion of the dwell time distributions of the different types of cargo. The only indications which are given are figures for the throughput-speed (number of times the storage area is used per year) and for the average stored quantity of goods (as a percentage of the annual throughput). These figures have been translated into a input-figures for the dwell time distributions of the four cargo types.

According to the report the inter-arrivaltime of ships is 16.5 hours (= 0.688 days). This means that the annual amount of ships is (365*0.688=) 531. HASPORT-II has separate generators for breakbulk-ships (breakbulk, wood, iron/steel/non-ferro) and for container-ships (containers/roro). On the basis of the throughput-figures and of the assumption that the average consignment-size of both ships (in tons) is equal, these 531 ships can be divided into (531*270000/1060000=) 135 container-ships and (531-135=) 396 breakbulk-ships. This results in inter-arrivaltimes of respectively 2.704 days and 0.922 days. Of the breakbulk-ships, 29.7% carries iron/steel/non-ferro, 30.4% carries wood and 39.9% carries general cargo; this results in inter-arrivaltimes of respectively 3.103 days, 3.032 days and 2.310 days.

The average consignment size of ships is (1060000/531) = 2000 tons. This is equally divided into 1000 tons import and 1000 tons export. For container-ships (with one container weighing 20 tons) this results in an average of 50 import-containers and 50 export-containers.

The transhipment of the cargo is either directly or indirectly (via intermediate storage on shore). For this model the direct transhipment takes place with barges and/or wagons. The required percentages for the input and the capacities of the trucks/wagons/barges have been determined as follows:

	Direct	Indirect	Trucks	Wagons	Barges
Gen. cargo	50 %	50 %	50 %	2 %	48 %
Wood	0 %	100%	98 %	2 %	0 %
Iron etc.	80 %	20 %	20 %	2 %	80 %
Cont./roro	0 %	100 %	100 %	0 %	0 %
Capacity for containers (TEU)			1.5	2	75
Capacity for breakbulk (tons)			10	30	200

4.4.4 Comparison of results

A simulation-run was performed using the input information as stated above. The results of the simulation of Waalhaven-Pier 2 using HASPORT-II and the calculations in the report are compared using output-information like storage area-requirements, berth-occupation-rates, annual throughputs, numbers of trucks, wagons and barges. The simulation resulted in the following figures for mean waiting-time and mean total port-time (= time between arriving at and leaving the port):

	Mean waiting-time	Mean total port-time
Container-ships		
- Class 1	4.87 hours	16.30 hours
Breakbulk-ships		
- Class 2	8.41 hours	58.28 hours
- Class 3	8.19 hours	57.97 hours
- Class 4	8.49 hours	58.12 hours

There are not much figures in the report to compare these waiting-time and port-time figures with. The mean total port time compares well with the value mentioned in the report: 57 hours.

A good comparison can be made with the report for the storage area requirement. In the report the gross area-requirement A of one type of cargo on the quay-side (open) or in a shed/warehouse (covered) is calculated in the following way:

$$A = T * p1 * p2 * f1 * f2$$

in which:

T	= annual throughput in tons
p1	= percentage, representing the average quantity of cargo which is stored
p2	= percentage of throughput which stored open or covered
f1	= peak-factor
f2	= gross package-factor in m2/ton

The results from the report and the HASPORT-II are as follows:

Storage area	Report		Simulation	
A (m^2)	open	covered	open	covered
Gen. cargo	3990	5740	4167	5931
Wood	43080	7600	43149	7632
Iron etc.	1220	960	1170	970
Containers/roro	4760	-	5293	-

The results of the which are used to make the comparison are the 95%-values of the frequency distributions of the storage requirements of the four cargotypes. The histograms of the distributions are shown in Figures A4.7 to A4.10.

The inter-arrivaltime of ships in the simulation-run is 0.685 days; it is close enough to 0.688

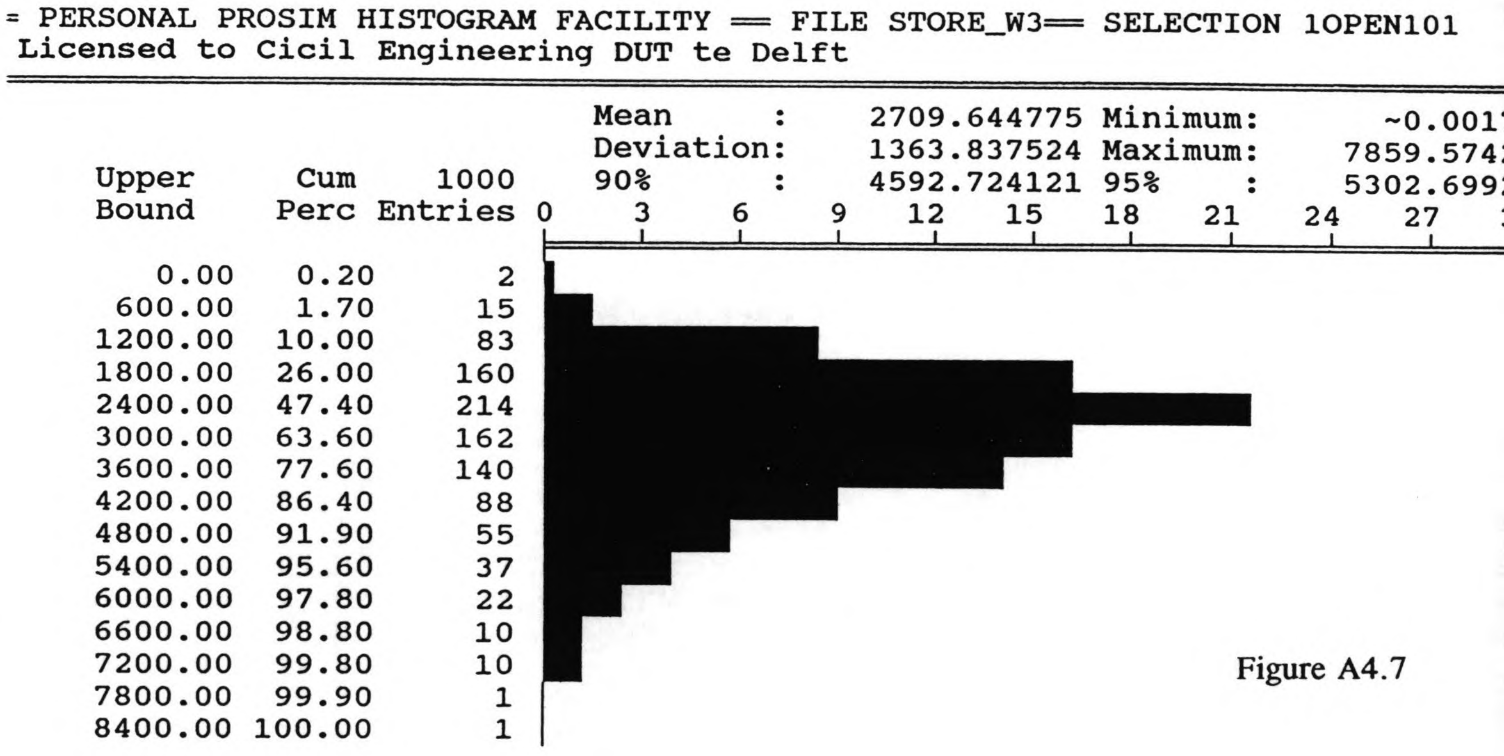
days. A check of the throughput-figures shows the following results:

Throughput (* 1000 tons)	Report	Simulation
Gen. cargo	315	317
Wood	240	246
Iron etc.	235	211
Containers/roro	270	284
TOTAL	1060	1058

The total throughput is only a little smaller than the forecasted throughput in the report.

The accuracy of the output information on the annual number of trucks, wagons and barges, depends mainly on the accuracy of the assumed values of the capacities of these means of transport. Therefore, comparison is quite difficult. The results are as follows:

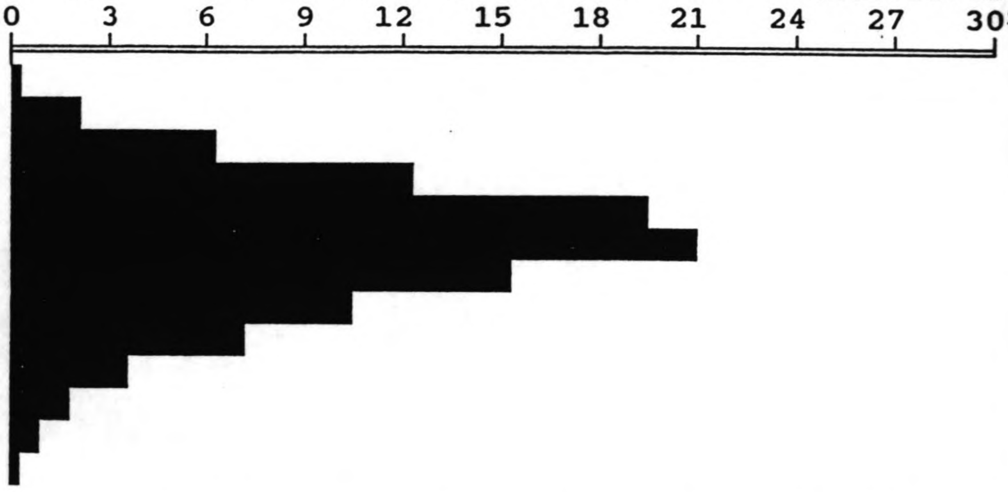
Number of:	Report	Simulation	
	per day	per year	per day
Trucks	158	54186	148
Wagons	1	353	1
Barges	no information	1586	4



```
= PERSONAL PROSIM HISTOGRAM FACILITY == FILE STORE_W3== SELECTION 1OPEN101
 Licensed to Cicil Engineering DUT te Delft

                                 Mean      :     2709.644775 Minimum:          ~0.001
                                 Deviation:      1363.837524 Maximum:         7859.574
   Upper       Cum      1000     90%       :     4592.724121 95%     :        5302.699
   Bound       Perc  Entries  0     3     6     9    12    15    18    21    24    27

     0.00      0.20        2
   600.00      1.70       15
  1200.00     10.00       83
  1800.00     26.00      160
  2400.00     47.40      214
  3000.00     63.60      162
  3600.00     77.60      140
  4200.00     86.40       88
  4800.00     91.90       55
  5400.00     95.60       37
  6000.00     97.80       22
  6600.00     98.80       10
  7200.00     99.80       10
  7800.00     99.90        1
  8400.00    100.00        1
```

Figure A4.7

```
 PERSONAL PROSIM HISTOGRAM FACILITY ═ FILE STORE_W3═ SELECTION 1OPEN201
Licensed to Cicil Engineering DUT te Delft

                              Mean      :     2317.594971 Minimum:        0.00129
                              Deviation:     1010.282898 Maximum:     5692.54150
 Upper      Cum      1000     90%       :     3743.057129 95%     :     4180.55761
 Bound      Perc  Entries 0    3    6    9    12   15   18   21   24   27   30

    0.00    0.20       2
  500.00    2.10      19
 1000.00    8.40      63
 1500.00   20.70     123
 2000.00   40.20     195
 2500.00   61.00     208
 3000.00   76.10     151
 3500.00   86.50     104
 4000.00   93.70      72
 4500.00   97.30      36
 5000.00   98.90      16
 5500.00   99.80       9
 6000.00  100.00       2
```



```
 PERSONAL PROSIM HISTOGRAM FACILITY ═ FILE STORE_W3═ SELECTION 1COV201
Licensed to Cicil Engineering DUT te Delft

                              Mean      :     3342.940430 Minimum:        0.00175
                              Deviation:     1402.623169 Maximum:     7661.08593
 Upper      Cum      1000     90%       :     5294.645020 95%     :     5909.09375
 Bound      Perc  Entries 0    2    4    6    8    10   12   14   16   18   20%

    0.00    0.20       2
  500.00    0.30       1
 1000.00    2.10      18
 1500.00    8.10      60
 2000.00   16.80      87
 2500.00   31.20     144
 3000.00   44.40     132
 3500.00   57.50     131
 4000.00   70.70     132
 4500.00   79.60      89
 5000.00   86.70      71
 5500.00   92.30      56
 6000.00   95.60      33
 6500.00   97.70      21
 7000.00   99.20      15
 7500.00   99.80       6
 8000.00  100.00       2
```

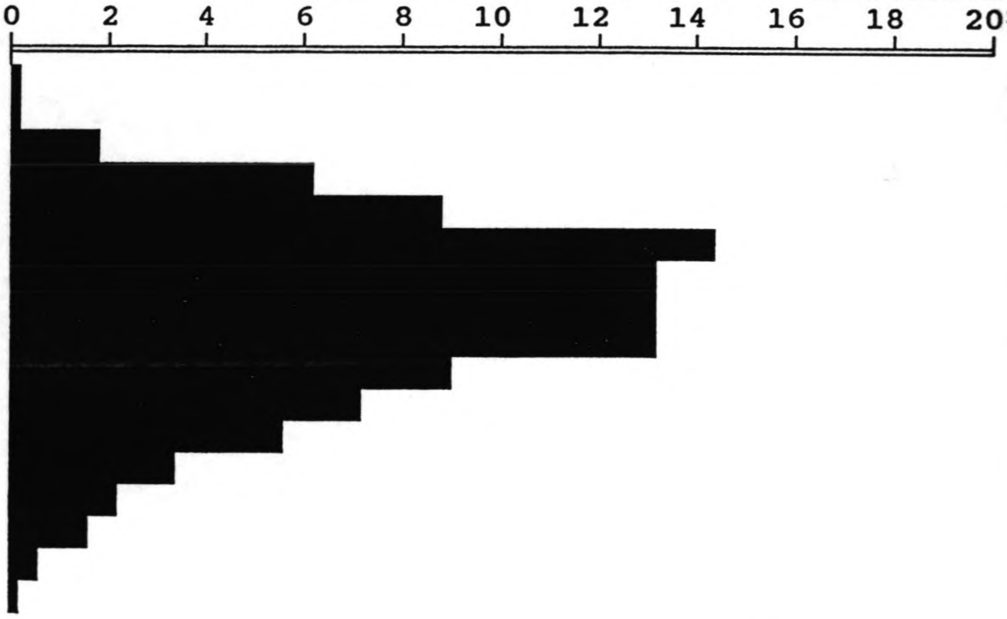
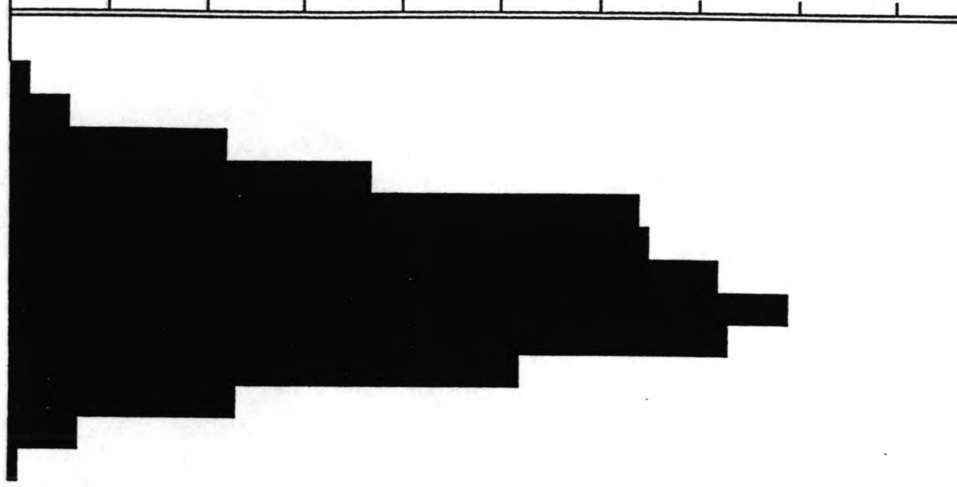


Figure A4.8

```
= PERSONAL PROSIM HISTOGRAM FACILITY ══ FILE STORE_W3══ SELECTION 1OPEN202
 Licensed to Cicil Engineering DUT te Delft

                                Mean      :   28558.617188 Minimum:     3091.894
                                Deviation:     9140.236328 Maximum:    53225.351
   Upper      Cum     1000      90%       :   40446.601562 95%     :   42888.882
   Bound      Perc  Entries  0     2     4     6     8    10    12    14    16    18

    2000.0    0.00        0
    6000.0    0.40        4
   10000.0    1.60       12
   14000.0    6.00       44
   18000.0   13.30       73
   22000.0   26.00      127
   26000.0   38.90      129
   30000.0   53.30      144
   34000.0   69.10      158
   38000.0   83.70      146
   42000.0   94.00      103
   46000.0   98.50       45
   50000.0   99.90       14
   54000.0  100.00        1
```



```
= PERSONAL PROSIM HISTOGRAM FACILITY ══ FILE STORE_W3══ SELECTION 1COV202
 Licensed to Cicil Engineering DUT te Delft

                                Mean      :    5099.706543 Minimum:      540.427
                                Deviation:     1625.023071 Maximum:     9607.302
   Upper      Cum     1000      90%       :    7238.823242 95%     :     7650.587
   Bound      Perc  Entries  0     2     4     6     8    10    12    14    16    18

      0.00    0.00        0
    700.00    0.40        4
   1400.00    1.20        8
   2100.00    3.10       19
   2800.00    8.10       50
   3500.00   17.60       95
   4200.00   30.50      129
   4900.00   43.70      132
   5600.00   60.10      164
   6300.00   74.60      145
   7000.00   87.10      125
   7700.00   95.60       85
   8400.00   99.00       34
   9100.00   99.90        9
   9800.00  100.00        1
```

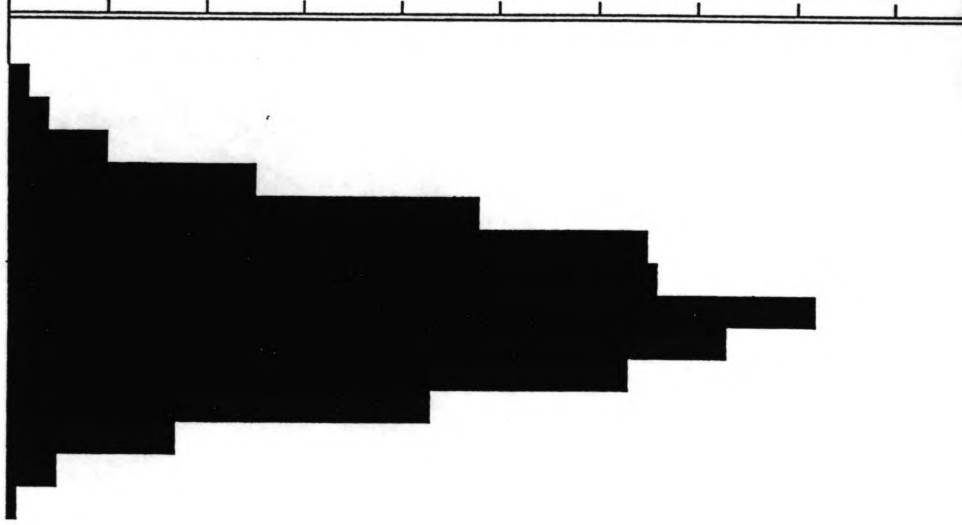
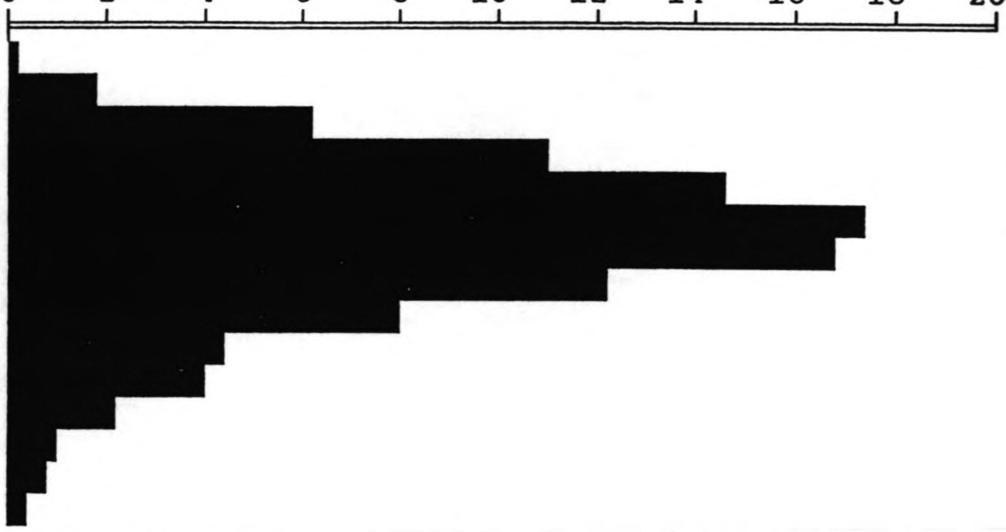


Figure A4.9

```
PERSONAL PROSIM HISTOGRAM FACILITY ══ FILE STORE_W3══ SELECTION 1OPEN203
icensed to Cicil Engineering DUT te Delft

                                  Mean      :       717.327148 Minimum:        181.969177
                                  Deviation:        243.406235 Maximum:       1550.512329
  Upper      Cum      1000        90%       :      1060.465210 95%     :      1182.500122
  Bound      Perc  Entries  0    2    4    6    8    10   12   14   16   18   20%

   200.00    0.20       2
   300.00    1.90      17
   400.00    7.90      60
   500.00   18.80     109
   600.00   33.30     145
   700.00   50.60     173
   800.00   67.20     166
   900.00   79.40     122
  1000.00   87.40      80
  1100.00   91.70      43
  1200.00   95.70      40
  1300.00   97.90      22
  1400.00   98.90      10
  1500.00   99.70       8
  1600.00  100.00       3
```



```
PERSONAL PROSIM HISTOGRAM FACILITY ══ FILE STORE_W3══ SELECTION 1COV203
icensed to Cicil Engineering DUT te Delft

                                  Mean      :       583.361389 Minimum:        159.909500
                                  Deviation:        199.293747 Maximum:       1281.426880
  Upper      Cum      1000        90%       :       859.322021 95%     :       978.787842
  Bound      Perc  Entries  0    3    6    9    12   15   18   21   24   27   30%

   100.00    0.00       0
   200.00    0.60       6
   300.00    5.70      51
   400.00   19.70     140
   500.00   38.20     185
   600.00   58.40     202
   700.00   75.40     170
   800.00   86.50     111
   900.00   92.40      59
  1000.00   95.70      33
  1100.00   98.50      28
  1200.00   99.50      10
  1300.00  100.00       5
```

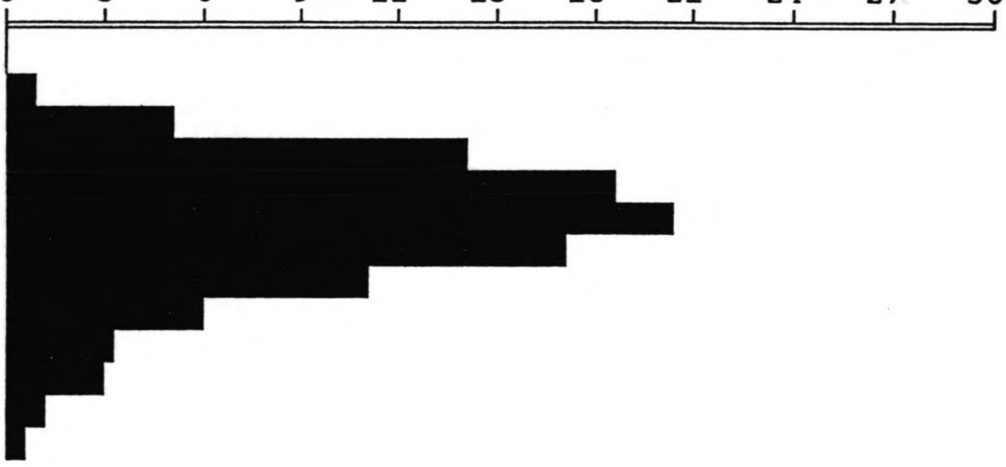


Figure A4.10

Annex 5 Details of the application of the model

This annex contains information of the input data and the output data of all simulation runs which have been performed for the Main Public Port in Pontianak and which have been used for the analysis in Chapter 7 of Volume 1. The first paragraph concerns the short term development, the second paragraph deals with the medium term development. All simulation runs have been given a reference number, which is indicated by a code between square parentheses ([...]). The first character of the code indicates the number of quays.

Run [5\1\20] is the calibrated run for the 1991-throughput-level. This run is the starting point for the short development analysis. Run [7\3\2] is the starting point for the medium term development analysis. Specific run characteristics of other medium term runs which differ from run [7\3\2] are mentioned separately.

The input and output data of the vessels are specified for each of the six shipclasses. These classes are:

* Class 1 = local vessels
* Class 2 = general cargo carrying interinsular vessels
* Class 3 = bagged cargo carrying interinsular vessels
* Class 4 = general cargo carrying ocean going vessels
* Class 5 = container carrying ocean going vessels
* Class 6 = passenger vessels

5.1 Simulation runs, short term development

RUN NR.	CHARACTERISTICS OF THE RUN
[5-1-20]	1991, 5 quays, no cargo handling improvement
[5-3-1]	1992, 5 quays, no cargo handling improvement
[5-4-1]	1993, 5 quays, no cargo handling improvement
[5-5-1]	1991, 5 quays, guaranteed cargo handling improvement
[5-6-1]	1992, 5 quays, guaranteed cargo handling improvement
[5-7-1]	1993, 5 quays, guaranteed cargo handling improvement
[5-8-1]	1994, 5 quays, guaranteed cargo handling improvement
[5-9-1]	1995, 5 quays, guaranteed cargo handling improvement
[5-10-1]	1996, 5 quays, guaranteed cargo handling improvement
[5-11-1]	1993, 5 quays, minimum cargo handling improvement
[5-12-1]	1994, 5 quays, minimum cargo handling improvement
[5-13-1]	1995, 5 quays, minimum cargo handling improvement
[5-14-1]	1996, 5 quays, minimum cargo handling improvement
[5-16-1]	1996, 5 quays, guaranteed cargo handling improvement, local vessels are also allocated to quay 3
[5-17-1]	1996, 5 quays, minimum cargo handling improvement, local vessels are also allocated to quay 3
[6-1-1]	1996, 6 quays, guaranteed cargo handling improvement
[6-2-1]	1996, 6 quays, minimum cargo handling improvement

Table A5.1
SHORT TERM DEVELOPMENT (1991-1996), CARGO AND VESSEL TRAFFIC

(N = annual number of vessels; IAT = mean interarrival time)

		class 1	class 2	class 3	class 4	class 5	class 6
1991	load, incoming	175	426	426	426	46	x
	load, outgoing	25	54	54	54	434	x
	mean total load	200	480	480	480	480	x
	N	793	422	388	145.6	13.4	180
	IAT (1/days)	0.460277	0.864929	0.940722	2.506868	27.23881	2.027778
1992	load, incoming	175	433.11	433.11	433.11	46.77	x
	load, outgoing	25	54.9	54.9	54.9	441.25	x
	mean total load	200	488.01	488.01	488.01	488.02	x
	N	846.61	443.13	407.72	150.71	16.25	191.21
	IAT (1/days)	0.431131	0.823686	0.895222	2.42187	22.46154	1.908896
1993	load, incoming	175	440.34	440.34	440.34	47.55	x
	load, outgoing	25	55.82	55.82	55.82	448.62	x
	mean total load	200	496.16	496.16	496.16	496.17	x
	N	903.84	465.31	427.82	155.62	19.7	203.12
	IAT (1/days)	0.403833	0.784423	0.853163	2.345457	18.52792	1.796967
1994	load, incoming	175	447.69	447.69	447.69	48.34	x
	load, outgoing	25	56.75	56.75	56.75	456.11	x
	mean total load	200	504.44	504.44	504.44	504.45	x
	N	964.54	488.61	449.24	160.21	23.89	215.77
	IAT (1/days)	0.378419	0.747017	0.812483	2.27826	15.27836	1.691616
1995	load, incoming	175	455.17	455.17	455.17	49.15	x
	load, outgoing	25	56.7	56.7	56.7	463.73	x
	mean total load	200	511.87	511.87	511.87	512.88	x
	N	1029.74	513.07	471.73	164.32	28.97	229.21
	IAT (1/days)	0.354458	0.711404	0.773748	2.221276	12.59924	1.592426
1996	load, incoming	175	462.77	462.77	462.77	49.97	x
	load, outgoing	25	58.66	58.66	58.66	471.47	x
	mean total load	200	521.43	521.43	521.43	521.44	x
	N	1099.35	539.81	495.35	168.21	35.15	243.49
	IAT (1/days)	0.332014	0.676164	0.736853	2.169907	10.38407	1.499035

Table A5.2
SHORT TERM DEVELOPMENT (1991-1996), RESULTS

No improvement / 5 quays

Year:		1991	1992	1993	1994	1995	1996
Run nr.:		[5\1\20]	[5\3\1]	[5\4\1]			
Average waiting time							
(hours)	class 1	68.57	309.88	1124.57	x	x	x
	class 2	38.22	91.74	433.6	x	x	x
	class 3	38.82	94.54	435	x	x	x
	class 4	41.35	95.49	438.06	x	x	x
	cl.2/3/4	38.95165	93.469	434.8482	x	x	x
	class 5	5.22	8.54	10.28	x	x	x
	class 6	10.08	11.87	13.69	x	x	x
Berth occupancy rate							
	berth 1	0.813	0.848	0.848	x	x	x
	berth 2	0.802	0.856	0.861	x	x	x
	b.1/2	0.808658	0.851158	0.853132	x	x	x
	b. 3/4/5	0.765	0.814	0.849	x	x	x
Storage area occ. rate							
	mean	0.357	0.466	0.918	x	x	x
	95 %	0.604	0.737	1.408	x	x	x

Guaranteed improvement / 5 quays

								local vessels at berth 3		new quay (berth 6)
Year:		1991	1992	1993	1994	1995	1996	1996		1996
Run nr.:		[5\5\1]	[5\6\1]	[5\7\1]	[5\8\1]	[5\9\1]	[5\10\1]	[5\16\1]		[6\1\1]
Average waiting time										
(hours)	class 1	12.97	18.12	27.08	45.59	150.9	892.47	38.83		3.35
	class 2	3.32	4.48	6.13	10.71	16.69	25.32	132.09		30.04
	class 3	3.03	4.33	6.08	10.69	18.01	26.48	134.01		31.49
	class 4	3.99	4.64	6.7	11.34	18.42	27.72	130.4		31.76
	cl.2/3/4	3.302878	4.441937	6.193778	10.79324	17.49366	26.14481	132.6637		30.89209
	class 5	1.78	1.81	4.7	3.19	4.75	5.8	7.11		6.41
	class 6	3.09	3.69	4.38	5.46	7	7.94	9.89		8.56
Berth occupancy rate										
	berth 1	0.701	0.733	0.764	0.802	0.838	0.838	0.806	berth 1	0.7
	berth 2	0.56	0.618	0.703	0.772	0.851	0.852	0.786	berth 2	0.566
									berth 3	0.394
	b.1/2	0.645342	0.687605	0.739921	0.790158	0.843132	0.843526	0.798105	b.1/2/3	0.550645
	b.3/4/5	0.529	0.566	0.607	0.641	0.692	0.734	0.804	b.4/5/6	0.751
Storage area occ. rate										
	mean	0.31	0.335	0.36	0.387	0.457	0.718	0.546		0.407
	95 %	0.546	0.587	0.626	0.682	0.764	1.113	0.931		0.708

Minimum improvement / 5 quays								local vessels at berth 3		new quay (berth 6)
Year:		1991	1992	1993	1994	1995	1996	1996		1996
Run nr.:				[5\11\1]	[5\12\1]	[5\13\1]	[5\14\1]	[5\17\1]		[6\2\1]
Average waiting time							890.27			
(hours)	class 1	x	x	26.91	45.18	146.63	885	8.84		3.35
	class 2	x	x	1.38	2.21	3.06	3.87	19.05		4.26
	class 3	x	x	1.18	1.98	2.91	3.85	18.55		4.33
	class 4	x	x	1.73	2.25	3.39	4.07	21.61		5.22
	cl.2/3/4	x	x	1.348515	2.119413	3.043578	3.88952	19.19693		4.423722
	class 5	x	x	0.7	1.5	1.64	1.46	4.85		1.37
	class 6	x	x	1.61	2.49	2.5	3.18	6.27		0.51
Berth occupancy rate										
	berth 1	x	x	0.759	0.794	0.836	0.838	0.741	berth 1	0.699
	berth 2	x	x	0.697	0.768	0.839	0.851	0.645	berth 2	0.571
									berth 3	0.392
	b.1/2	x	x	0.734526	0.783737	0.837184	0.843132	0.703105	b.1/2/3	0.55073
	b.3/4/5	x	x	0.448	0.474	0.503	0.538	0.667	b.4/5/6	0.531
Storage area occ. rate										
	mean	x	x	0.311	0.332	0.387	0.613	0.393		0.383
	95 %	x	x	0.541	0.583	0.646	0.944	0.695		0.673

Cargo throughput

		1991	1992	1993	1994	1995	1996
ton*1000							
	class 1	160	170	183	195	207	223
	class 2	202	218	232	246	260	277
	class 3	191	203	216	228	248	262
	class 4	76	80	82	84	87	91
	class 5	5	6	8	9	13	17
	total	634	677	721	762	815	869

5.2 Simulation runs, medium term development

Simulation time: 850 days

RUN NR.	CHARACTERISTICS OF THE RUN
[7\3\2]	2002, medium throughput scenario, 7 quays, minimum cargo handling improvement no strikes, no bad weather mooring time = 0.5 hours; mooring restriction = 0.2 hours no deviation in mean dwell time
[7\3\3]	2002, medium throughput scenario, 7 quays, minimum cargo handling improvement deviation in mean dwell time
[7\4\2]	2002, medium throughput scenario, 7 quays, minimum cargo handling improvement strike (interval time = 365 days, duration = 2 days)
[7\5\3]	2002, medium throughput scenario, 7 quays, minimum cargo handling improvement bad weather (interval time = 120 days, duration = 1 days)
[7\6\1]	2002, medium throughput scenario, 7 quays, minimum cargo handling improvement strike (interval time = 365 days, duration = 2 days) bad weather (interval time = 120 days, duration = 1 days)
[7\7\1]	2002, medium throughput scenario, 7 quays, minimum cargo handling improvement mooring time = 1.0 hours
[7\7\2]	2002, medium throughput scenario, 7 quays, minimum cargo handling improvement mooring time = 0.75 hours
[7\8\1]	2002, medium throughput scenario, 7 quays, minimum cargo handling improvement mooring restriction = 0 hours
[7\8\2]	2002, medium throughput scenario, 7 quays, minimum cargo handling improvement mooring restriction = 0.4 hours

[7\low]	2002, low throughput scenario 7 quays, minimum cargo handling improvement
[7\lo2]	2002, semi-low throughput scenario 7 quays, minimum cargo handling improvement
[7\hi2]	2002, semi-high throughput scenario 7 quays, minimum cargo handling improvement
[7\hi]	2002, high throughput scenario 7 quays, minimum cargo handling improvement
[7g\low]	2002, low throughput scenario 7 quays, guaranteed cargo handling improvement
[7g\lo2]	2002, semi-low throughput scenario 7 quays, guaranteed cargo handling improvement
[7m\low]	2002, low throughput scenario 7 quays, medium cargo handling improvement
[7m\lo2]	2002, semi-low throughput scenario 7 quays, medium cargo handling improvement
[7m]	2002, medium throughput scenario 7 quays, medium cargo handling improvement
[7m\hi2]	2002, semi-high throughput scenario 7 quays, medium cargo handling improvement
[7m\hi]	2002, high throughput scenario 7 quays, medium cargo handling improvement
[8\1\low]	2002, low throughput scenario 8 quays, minimum cargo handling improvement
[8\1\lo2]	2002, semi-low throughput scenario 8 quays, minimum cargo handling improvement
[8\1\2]	2002, medium throughput scenario, 8 quays, minimum cargo handling improvement
[8\1\hi2]	2002, semi-high throughput scenario 8 quays, minimum cargo handling improvement
[8\1\hi]	2002, high throughput scenario 8 quays, minimum cargo handling improvement

[8\2\low]	2002, low throughput scenario
	8 quays, guaranteed cargo handling improvement
[8\2\lo2]	2002, semi-low throughput scenario
	8 quays, guaranteed cargo handling improvement
[8\2\1]	2002, medium throughput scenario,
	8 quays, guaranteed cargo handling improvement
[8\2\hi2]	2002, semi-high throughput scenario
	8 quays, guaranteed cargo handling improvement
[8\2\hi]	2002, high throughput scenario
	8 quays, guaranteed cargo handling improvement
[6\3\1]	2002, medium throughput scenario,
	6 quays, minimum cargo handling improvement
[6m]	2002, medium throughput scenario
	6 quays, medium cargo handling improvement
[6m\hi2]	2002, semi-high throughput scenario
	6 quays, medium cargo handling improvement
[6m\hi]	2002, high throughput scenario
	6 quays, medium cargo handling improvement
[7\3\4]	2002, medium throughput scenario,
	7 quays, minimum cargo handling improvement
	as [7/3/2], but simulation time = 400 days
[7\3\5]	2002, medium throughput scenario,
	7 quays, minimum cargo handling improvement
	as [7/3/2], but simulation time = 400 days and different seeds
[7\3\6]	2002, medium throughput scenario,
	7 quays, minimum cargo handling improvement
	as [7/3/2], but simulation time = 400 days and different seeds
[7\3\7]	2002, medium throughput scenario,
	7 quays, minimum cargo handling improvement
	as [7/3/2], but simulation time = 400 days and different seeds
[7\3\8]	2002, medium throughput scenario,
	7 quays, minimum cargo handling improvement
	as [7/3/2], but simulation time = 400 days and different seeds
[7\3\9]	2002, medium throughput scenario,
	7 quays, minimum cargo handling improvement
	as [7/3/2], but simulation time = 400 days and different seeds

QUAY CONFIGURATIONS / QUAY ALLOCATION RULES:

7 quays:				
*	Quay 2	100 m	max. 3 vessels	Class 1
*	Quays 3&4	217 m	max. 6 vessels	Class 1, Class 5
*	Quays 5&6	195 m	max. 3 vessels	Class 1, Class 2, Class 3, Class 4
*	Quay 7	120 m	max. 2 vessels	Class 2, Class 3, Class 4
(*	Quay 1	115 m	max. 1 vessel	Class 6)
8 quays:				
*	Quay 2	100 m	max. 3 vessels	Class 1
*	Quays 3&4	217 m	max. 6 vessels	Class 1, Class 5
*	Quays 5&6	195 m	max. 3 vessels	Class 1, Class 2, Class 3, Class 4
*	Quays 7&8	190 m	max. 3 vessels	Class 2, Class 3, Class 4
(*	Quay 1	115 m	max. 1 vessel	Class 6)
6 quays:				
*	Quay 2	100 m	max. 3 vessels	Class 1
*	Quays 3&4	217 m	max. 6 vessels	Class 1, Class 5
*	Quays 5&6	195 m	max. 3 vessels	Class 2, Class 3, Class 4
(*	Quay 1	115 m	max. 1 vessel	Class 6)

CARGO HANDLING SCENARIOS:	Rate per hour at berth(ton)	Number hours per day	Number of gangs per ves.	Net ouput per gang (ton/h)	Rate of ships gear (ton/h)
Guaranteed improvement:					
* Class 1/gen. cargo:	5.0	8	1	15.0	x
* Class 2&4/gen. cargo:	20.0	8	3	20.0	x
* Class 3/bag. cargo:	20.0	8	3	20.0	x
* Class 5/containers:	104.2	8	2	15.0	74.2
Minimum improvement:					
* Class 1/gen. cargo:	5.0	8	1	15.0	x
* Class 2&4/gen. cargo:	23.1	8	3	23.1	x
* Class 3/bag. cargo:	40.4	8	3	40.4	x
* Class 5/containers:	104.2	8	2	15.0	74.2
Medium improvement:					
* Class 1/gen. cargo:	10.0	16	1	15.0	x
* Class 2&4/gen. cargo:	37.0	16	3	18.5	x
* Class 3/bag. cargo:	82.0	16	3	41.0	x
* Class 5/containers:	104.2	8	2	15.0	74.2

Table A5.3
MEDIUM TERM DEVELOPMENT (2002), CARGO AND VESSEL TRAFFIC

(N = annual number of vessels; IAT = mean inter arrival time)

Throughput scenario		class 1	class 2	class 3	class 4	class 5	class 6
Low	load, incoming	172.36	496.4	496.4	496.4	198.6	x
	load, outgoing	27.64	79.6	79.6	79.6	377.4	x
	mean total load	200	576	576	576	576	x
	N	1785.49	578.72	537.3	105.59	121.64	315
	IAT (1/days)	0.204426	0.630702	0.679323	3.456767	3.000658	1.15873
Low2	load, incoming	172.36	496.4	496.4	496.4	198.6	x
	load, outgoing	27.64	79.6	79.6	79.6	377.4	x
	mean total load	200	576	576	576	576	x
	N	1921.18	622.39	577.85	113.56	130.82	332.5
	IAT (1/days)	0.189987	0.586449	0.631652	3.21416	2.790093	1.097744
Medium	load, incoming	172.36	496.4	496.4	496.4	198.6	x
	load, outgoing	27.64	79.6	79.6	79.6	377.4	x
	mean total load	200	576	576	576	576	x
	N	2055	666.07	618.4	121.53	140	350
	IAT (1/days)	0.177616	0.54799	0.590233	3.003374	2.607143	1.042857
High2	load, incoming	172.36	496.4	496.4	496.4	198.6	x
	load, outgoing	27.64	79.6	79.6	79.6	377.4	x
	mean total load	200	576	576	576	576	x
	N	2189.75	709.75	658.95	129.5	149.18	367.5
	IAT (1/days)	0.166686	0.514266	0.553912	2.818533	2.446709	0.993197
High	load, incoming	172.36	496.4	496.4	496.4	198.6	x
	load, outgoing	27.64	79.6	79.6	79.6	377.4	x
	mean total load	200	576	576	576	576	x
	N	2342.5	753.42	699.5	137.49	158.36	385
	IAT (1/days)	0.155816	0.484458	0.521801	2.654739	2.304875	0.948052

Table A5.4
MEDIUM TERM DEVELOPMENT (2002), VESSEL PERFORMANCES

(wait = mean waiting time in hours)
(berth = mean time at berth in hours)
(ratio = waiting time / time at berth)

7 BERTHS			class 1	class 2	class 3	cl.2\3	class 4	cl.2\3\4	class 5
strike	[7\3\2]	wait	20.44	35.38	36.4	35.8831	41.42	36.3439	12.81
	[7\4\2]	wait	23.84	39.42	39.61	39.5137	45.64	40.0236	13.73
	[7\3\2]	berth	41.31	26.06	14.96	20.5852	28.54	21.2472	15.35
	[7\4\2]	berth	41.74	26.23	15.18	20.7798	28.42	21.4157	15.56
	[7\3\2]	ratio	0.4948	1.35764	2.43316	1.94429	1.4513	1.71053	0.83453
	[7\4\2]	ratio	0.57115	1.50286	2.60935	1.90154	1.60591	1.86889	0.88239
typhoon	[7\3\2]	wait	20.44	35.38	36.4	35.8831	41.42	36.3439	12.81
	[7\5\3]	wait	32.4	48.55	50.49	49.5069	59.07	50.3028	13.96
	[7\3\2]	berth	41.31	26.06	14.96	20.5852	28.54	21.2472	15.35
	[7\5\3]	berth	41.79	26.34	15.18	20.8356	28.91	21.5076	15.86
	[7\3\2]	ratio	0.4948	1.35764	2.43316	1.74315	1.4513	1.71053	0.83453
	[7\5\3]	ratio	0.77531	1.8432	3.32609	2.37608	2.04324	2.33884	0.8802
strike+	[7\3\2]	wait	20.44	35.38	36.4	35.8831	41.42	36.3439	12.81
typhoon	[7\6\1]	wait	37.92	56.63	57.47	57.0443	64.46	57.6615	14.18
	[7\3\2]	berth	41.31	26.06	14.96	20.5852	28.54	21.2472	15.35
	[7\6\1]	berth	42.05	26.48	15.23	20.9312	29.3	21.6277	15.52
	[7\3\2]	ratio	0.4948	1.35764	2.43316	1.74315	1.4513	1.71053	0.83453
	[7\6\1]	ratio	0.90178	2.1386	3.77347	2.72533	2.2	2.6661	0.91366
through-	[7\3\low]	wait	5.18	9.47	9.81	9.63795	12.25	9.85369	9.44
put	[7\3\lo2]	wait	9.05	16.38	17.08	16.7244	20.78	17.062	9.48
	[7\3\2]	wait	20.44	35.38	36.4	35.8831	41.42	36.3467	12.81
	[7\3\hi2]	wait	131	153.11	155.44	154.257	165.11	155.172	14.37
	[7\3\hi]	wait	247.99	272.7	269.34	271.081	280.48	271.846	15.21
	[7\3\low]	berth	40.49	25.46	15.06	20.3226	27.52	20.917	16.18
	[7\3\lo2]	berth	40.74	25.98	14.8	20.4787	28.6	21.1572	15.75
	[7\3\2]	berth	41.31	26.06	14.96	20.5852	28.54	21.2333	15.35
	[7\3\hi2]	berth	41.82	26.25	14.8	20.6118	27.69	21.2052	15.97
	[7\3\hi]	berth	40.6	26.54	14.69	20.8288	30.37	21.5275	16.78
	[7\3\low]	ratio	0.12793	0.37196	0.65139	0.47425	0.44513	0.47108	0.58344
	[7\3\lo2]	ratio	0.22214	0.63048	1.15405	0.81667	0.72657	0.80644	0.6019
	[7\3\2]	ratio	0.4948	1.35764	2.43316	1.74315	1.4513	1.71178	0.83453
	[7\3\hi2]	ratio	3.13247	5.83276	10.5027	7.48393	5.9628	7.31762	0.89981
	[7\3\hi]	ratio	6.10813	10.2751	18.3349	13.0147	9.23543	12.6278	0.90644

mooring	[7\3\2]	wait	20.44	35.38	36.4	35.8831	41.42	36.3439	12.81
time	[7\7\2]	wait	29.86	45.77	46.88	46.3175	55.32	47.0667	13.94
	[7\7\1]	wait	50.21	72.41	73.42	72.9082	84.32	73.8579	12.64
	[7\3\2]	berth	41.31	26.06	14.96	20.5852	28.54	21.2472	15.35
	[7\7\2]	berth	41.43	26.29	14.87	20.6573	28.6	21.3184	16.32
	[7\7\1]	berth	41.68	26.53	15.25	20.9664	27.76	21.5318	15.94
	[7\3\2]	ratio	0.4948	1.35764	2.43316	1.74315	1.4513	1.71053	0.83453
	[7\7\2]	ratio	0.72073	1.74097	3.15266	2.24218	1.93427	2.2078	0.85417
	[7\7\1]	ratio	1.20465	2.72936	4.81443	3.47739	3.03746	3.43018	0.79297
mooring	[7\8\1]	wait	14.95	24.42	25.95	25.1746	31.25	25.6803	12.03
restr.	[7\3\2]	wait	20.44	35.38	36.4	35.8831	41.42	36.3439	12.81
	[7\8\2]	wait	65.63	92.78	93.61	93.1894	105.31	94.1981	14.68
	[7\8\1]	berth	40.63	25.41	14.23	19.8957	28.02	20.5719	15.02
	[7\3\2]	berth	41.31	26.06	14.96	20.5852	28.54	21.2472	15.35
	[7\8\2]	berth	42.98	27.26	16.22	21.8147	30.1	22.5043	16.84
	[7\8\1]	ratio	0.36795	0.96104	1.82361	1.26533	1.11527	1.24832	0.80093
	[7\3\2]	ratio	0.4948	1.35764	2.43316	1.74315	1.4513	1.71053	0.83453
	[7\8\2]	ratio	1.52699	3.40352	5.77127	4.27185	3.49867	4.18578	0.87173
medium	[7m\low]	wait	0.04	0.012	0.19	0.09993	0.28	0.1148	0.74
rate	[7m\lo2]	wait	0.07	0.28	0.22	0.25048	0.45	0.2671	0.87
	[7m]	wait	0.11	0.38	0.35	0.3652	0.73	0.39561	0.99
	[7m\hi2]	wait	0.14	0.52	0.49	0.50523	0.81	0.53089	1.09
	[7m\hi]	wait	0.21	0.66	0.66	0.66	1	0.68879	1.32
	[7m\low]	berth	20.26	15.98	7.53	11.8058	18.02	12.3191	8.33
	[7m\lo2]	berth	20.28	16.15	7.55	11.9182	17.61	12.3939	8.6
	[7m]	berth	20.48	16.14	7.57	11.913	17.17	12.3396	8.36
	[7m\hi2]	berth	20.48	16.05	7.57	11.8743	17.26	12.3259	8.17
	[7m\hi]	berth	20.46	16.09	7.55	11.9741	17.01	12.3218	8.43
	[7m\low]	ratio	0.00197	0.00075	0.02523	0.00846	0.01554	0.00932	0.08884
	[7m\lo2]	ratio	0.00345	0.01734	0.02914	0.02102	0.02555	0.02155	0.10116
	[7m]	ratio	0.00537	0.02354	0.04624	0.03066	0.04252	0.03206	0.11842
	[7m\hi2]	ratio	0.00684	0.0324	0.06473	0.04255	0.04693	0.04307	0.13341
	[7m\hi]	ratio	0.01026	0.04102	0.08742	0.05512	0.05879	0.0559	0.15658
guarant.	[7g\low]	wait	15.67	64.72	68.94	66.8046	82.74	68.1208	10.67
rate	[7g\lo2]	wait	64.43	162.35	163.55	162.94	203.43	166.312	13.64
	[7g\low]	berth	42.08	30.5	29.61	30.0604	33.19	30.3188	15.99
	[7g\lo2]	berth	42.45	30.9	29.74	30.3292	33.13	30.5627	15.85
	[7g\low]	ratio	0.37239	2.12197	2.32827	2.22235	2.49292	2.24681	0.66729
	[7g\lo2]	ratio	1.51779	5.25405	5.49933	5.3724	6.14036	5.44167	0.86057

8 BERTHS			class 1	class 2	class 3	cl.2\3	class 4	cl.2\3\4	class 5
through-	[8\1\lo]	wait	3.73	2.9	2.94	2.91976	2.89	2.9173	7.71
put	[8\1\lo2]	wait	6.17	5.13	5.54	5.33175	5.32	5.33069	9.19
	[8\1\2]	wait	9.99	9.95	10.78	10.3594	10.59	10.3798	10.61
	[8\1\hi2]	wait	19.57	20.08	21.16	20.6118	22.38	20.761	12.69
	[8\1\hi]	wait	104.24	103.74	105.06	104.376	108.61	104.747	15.41
	[8\1\lo]	berth	40.67	25.44	14.78	20.1741	28.07	20.8263	16.36
	[8\1\lo2]	berth	40.82	25.83	15.14	20.5698	27.89	21.1815	15.69
	[8\1\2]	berth	41.07	25.71	15.07	20.462	28.35	21.1052	15.6
	[8\1\hi2]	berth	41.42	25.62	14.83	20.3068	27.88	20.9421	16.02
	[8\1\hi]	berth	41.67	25.91	15.04	20.6711	27.62	21.1594	16.29
	[8\1\lo]	ratio	0.09171	0.11399	0.19892	0.14473	0.10296	0.14008	0.47127
	[8\1\lo2]	ratio	0.15115	0.19861	0.36592	0.2592	0.19075	0.25167	0.58572
	[8\1\2]	ratio	0.24324	0.38701	0.71533	0.50627	0.37354	0.49181	0.68013
	[8\1\hi2]	ratio	0.47248	0.78376	1.42684	1.01502	0.80273	0.99135	0.79213
	[8\1\hi]	ratio	2.50156	4.00386	6.98537	5.04938	3.9323	4.95037	0.94598
guarant.	[8\2\low]	wait	6.58	13.66	14.43	14.0404	14.81	14.1039	9.98
rate	[8\2\lo2]	wait	11.82	26.78	28.72	27.7346	31.6	28.0561	10.03
	[8\2\1]	wait	42.51	75.31	77.29	76.2866	81.6	76.733	13.77
	[8\2\hi2]	wait	313.8	349.85	350.83	350.333	341.21	349.564	14.21
	[8\2\low]	berth	41.07	30.05	29.31	29.6856	33.41	29.9922	16.37
	[8\2\lo2]	berth	41.38	30.81	29.5	30.1654	33.13	30.4125	15.91
	[8\2\1]	berth	41.92	30.95	29.64	30.3049	33	30.527	15.89
	[8\2\hi2]	berth	41.78	30.75	29.75	30.2576	34.65	30.6273	15.75
	[8\2\low]	ratio	0.16021	0.45458	0.49232	0.47297	0.44328	0.47025	0.60965
	[8\2\lo2]	ratio	0.28565	0.8692	0.97356	0.91942	0.95382	0.92252	0.63042
	[8\2\1]	ratio	1.01407	2.43328	2.60762	2.5173	2.47273	2.51361	0.86658
	[8\2\hi2]	ratio	7.51077	11.3772	11.7926	11.5784	9.84733	11.4135	0.90222

6 BERTHS			class 1	class 2	class 3	cl.2\3	class 4	cl.2\3\4	class 5
medium	[6m]	wait	0.76	5.32	5.08	5.20162	5.9	5.25958	1.51
rate	[6m\hi2]	wait	1	6.85	6.55	6.70203	7.61	6.77866	1.67
	[6m\hi]	wait	1.7	9.2	9.15	9.17534	10.46	9.28416	2.07
	[6m]	berth	20.49	16.12	7.48	11.8585	17.73	12.3362	8.1
	[6m\hi2]	berth	20.51	15.96	7.51	11.7922	17.41	12.2696	8.16
	[6m\hi]	berth	20.51	16.07	7.43	11.8085	16.8	12.2407	8.28
	[6m]	ratio	0.03709	0.33002	0.67914	0.43864	0.33277	0.42635	0.18642
	[6m\hi2]	ratio	0.04876	0.4292	0.87217	0.56834	0.43711	0.55247	0.20466
	[6m\hi]	ratio	0.08289	0.5725	1.23149	0.77701	0.62262	0.75847	0.25

Table A5.5
MEDIUM TERM DEVELOPMENT (2002), BERTH AND STORAGE AREA UTILIZATION

		Berth Occupancy Rates					Area Occ. Rate	
		Quay2	Quay3/4	Quay5/6	Quay7(/8)	Q.2-8	Mean	95%boun
7 BERTHS								
strike	[7\3\2]	0.87	0.831	0.815	0.819	0.82996	0.717	1.137
	[7\4\2]	0.874	0.836	0.818	0.824	0.83418	0.637	1.09
typhoon	[7\3\2]	0.87	0.831	0.815	0.819	0.82996	0.717	1.137
	[7\5\3]	0.879	0.84	0.822	0.833	0.83929	0.633	1.1
strike	[7\3\2]	0.87	0.831	0.815	0.819	0.82996	0.717	1.137
+typhoon	[7\6\1]	0.886	0.846	0.826	0.836	0.84426	0.557	1.077
through-	[7\3\low]	0.815	0.715	0.717	0.621	0.71359	0.591	0.981
put	[7\3\lo2]	0.836	0.771	0.771	0.718	0.77122	0.64	1.023
	[7\3\2]	0.87	0.831	0.815	0.819	0.82996	0.717	1.137
	[7\3\hi2]	0.927	0.887	0.851	0.877	0.88032	0.991	1.437
	[7\3\hi]	0.921	0.89	0.858	0.871	0.88142	1.374	1.989
storage	[7\3\2]	0.87	0.831	0.815	0.819	0.82996	0.717	1.137
deviat.	[7\3\3]	0.876	0.83	0.811	0.81	0.82762	0.717	1.117
mooring	[7\3\2]	0.87	0.831	0.815	0.819	0.82996	0.717	1.137
time	[7\7\2]	0.881	0.837	0.813	0.817	0.83276	0.737	1.159
	[7\7\1]	0.845	0.818	0.886	0.821	0.84382	0.779	1.203
mooring	[7\8\1]	0.864	0.811	0.796	0.792	0.81115	0.699	1.118
restr.	[7\3\2]	0.87	0.831	0.815	0.819	0.82996	0.717	1.137
	[7\8\2]	0.912	0.872	0.846	0.865	0.86898	0.814	1.244
medium	[7m\low]	0.649	0.344	0.425	0.151	0.38061	0.525	0.868
rate	[7m\lo2]	0.664	0.38	0.454	0.178	0.40941	0.563	0.945
	[7m]	0.679	0.413	0.48	0.22	0.43912	0.602	0.992
	[7m\hi2]	0.702	0.439	0.509	0.248	0.46595	0.643	1.024
	[7m\hi]	0.714	0.476	0.534	0.287	0.49567	0.687	1.083
guarant.	[7g\low]	0.863	0.796	0.873	0.878	0.84593	0.663	1.07
rate	[7g\lo2]	0.915	0.882	0.932	0.912	0.90834	0.844	1.294
8 BERTHS								
through-	[8\1\low]	0.813	0.707	0.667	0.458	0.64228	0.511	0.854
put	[8\1\lo2]	0.831	0.747	0.711	0.548	0.69407	0.549	0.899
	[8\1\2]	0.848	0.794	0.736	0.636	0.74206	0.6	0.968
	[8\1\hi2]	0.884	0.841	0.757	0.727	0.79247	0.661	1.026
	[8\1\hi]	0.922	0.884	0.774	0.827	0.84331	0.857	1.236
guarant.	[8\2\low]	0.824	0.735	0.778	0.695	0.74842	0.533	0.865
rate	[8\2\lo2]	0.855	0.802	0.826	0.783	0.81088	0.587	0.965
	[8\2\1]	0.907	0.873	0.877	0.846	0.87147	0.694	1.064
	[8\2\hi2]	0.937	0.896	0.896	0.861	0.89215	1.205	1.67
6 BERTHS								
medium	[6m]	0.694	0.445	0.57	x	0.54124	0.733	1.188
rate	[6m\hi2]	0.718	0.483	0.603	x	0.5746	0.786	1.269
	[6m\hi]	0.731	0.533	0.636	x	0.6109	0.842	1.307

Table A5.6
SEED CHANGE

VESSEL PERFORMANCES

			class 1	class 2	class 3	class 4	cl.2/3/4	class 5	total
[7/3/4]	wait	(mean)	22.5	38.76	37.87	44.27	38.8459	12.55	x
	(hours)	(st.dev.)	21.98	21.19	29.52	28.2	x	9.6	x
		(95% b.)	65.76	88.08	88.08	98.4	x	32.26	x
	N		2082	677	613	121	1411	135	3628
	thr.put	(1000ton)	413	400	344	79	823	80	1316
[7/3/5]	wait	(mean)	49.92	66.02	69.14	70.63	67.7328	13.3	x
	(hours)	(st.dev.)	43.68	44.83	45.36	50.66	x	11.16	x
		(95% b.)	127.68	140.4	140.16	151.56	x	34.99	x
	N		2034	708	625	115	1448	142	3624
	thr.put	(1000ton)	415	414	375	63	852	86	1353
[7/3/6]	wait	(mean)	13.8	24	23.95	24.55	24.0239	10.13	x
	(hours)	(st.dev.)	16.63	21.82	22.92	22.8	x	9.31	x
		(95% b.)	51.85	71	79.29	69	x	25.46	x
	N		2011	678	635	120	1433	123	3567
	thr.put	(1000ton)	399	382	347	61	790	61	1250
[7/3/7]	wait	(mean)	23.41	35.85	36.55	35.77	36.1535	9.91	x
	(hours)	(st.dev.)	27.12	35.52	35.52	36.72	x	11.76	x
		(95% b.)	81.6	109.92	110.4	105.8	x	33.12	x
	N		2037	667	613	106	1386	128	3551
	thr.put	(1000ton)	410	390	355	61	806	73	1289
[7/3/8]	wait	(mean)	21.95	32.2	35.29	34.19	33.7176	13.32	x
	(hours)	(st.dev.)	22.56	27.96	27.77	28.32	x	11.81	x
		(95% b.)	67.92	84.7	86.9	84.72	x	38.64	x
	N		2129	658	602	110	1370	140	3639
	thr.put	(1000ton)	417	381	339	66	786	83	1286
[7/3/9]	wait	(mean)	16.52	30.34	30.58	31.44	30.5566	10.88	x
	(hours)	(st.dev.)	18.96	26.09	27.38	28.51	x	11.11	x
		(95% b.)	57.6	83.86	85.44	91.68	x	35.21	x
	N		2054	611	632	133	1376	133	3563
	thr.put	(1000ton)	408	354	344	91	789	71	1268
Average	wait	(mean)	24.6833	37.8617	38.8967	40.1417	38.505	11.6817	x
	N		2057.83	666.5	620	117.5	1404	133.5	3595.33
	thr.put	(1000ton)	410.333	386.833	350.667	70.1667	807.667	75.6667	1293.67

BERTH AND STORAGE UTILIZATION

	Berth Occupancy Rates					Area Occ. Rate	
	Quay2	Quay3/4	Quay5/6	Quay7	Q.2-8	Mean	95%bound
[7\3\4]	0.894	0.848	0.834	0.827	0.84697	0.749	1.099
[7\3\5]	0.91	0.866	0.845	0.847	0.86288	0.767	1.112
[7\3\6]	0.864	0.799	0.792	0.798	0.80694	0.615	0.98
[7\3\7]	0.856	0.815	0.805	0.786	0.8129	0.69	1.17
[7\3\8]	0.881	0.848	0.815	0.811	0.83601	0.735	1.207
[7\3\9]	0.881	0.819	0.866	0.817	0.84293	0.704	1.1
Average	0.881	0.8325	0.82617	0.81433	0.83477	0.71	x

<u>Annex 6</u> <u>Listing of the program</u>

<u>Port simulation model HASPORT-II</u>

MODULES	Define
	Main
	Generator
	Harbour
	Terminal
	Crane
	Ship
	Shift
	Storage
	Strike
	Typhoon
	Report
	Passengership
MACROS	Patterns
	Arrivalpattern
	Departurepattern
	Patternchange
	Error-1
	Error-2
	Error-3
	DWT-tables

<u>Model extensions</u>

MODULES	Harbour
	Terminal
MACRO	Shiftships

```
MODEL HP2_11P                                               Date:  93/08/16
MOD DEFINE                                                  Time:  18:27:37

@@@@@@@@@@@@@@@@@@@@@@@
@  DEFINITION MODULE  @
@@@@@@@@@@@@@@@@@@@@@@@

CLASS                  : SHIP             SHIPCLASS
                         BERTH            CRANE
                         CARGOTYPE        TERMINAL
                         SHIFT            GENERATOR
                         ARRIVALDAY       DEPARTUREDAY
COMPONENT              : HARBOUR_MASTER   STORAGE_MASTER
                         TERMINAL_MASTER  REPORTER
                         TYPHOON          STRIKE
QUEUE                  : ARRIVINGSHIPS    DEPARTINGSHIPS
                         PORT             ROW
                         QUAY             BUOY
                         PRIORITY_ROW

RANDOMSTREAM           : NORM_I[9] NORM_E[9]   UNIF        UNIF_DWT[9]
                         EXP_IAT_T EXP_IAT_S   EXP_IAT_B   EXP_IAT[9]
                         UNKN_M[9] NORM_C[9]
                         NORM_TD   NORM_SD     NORM_BD     NORM_IAT[9]
INPUTSTREAM            : DFILE     HFILE       SFILE       TFILE
OUTPUTSTREAM           : REPORT    CHECKLIST
TIMEUNIT               : DAYS

ATTRIBUTES OF MAIN:
  REAL                 : MEAN_IAT_T       MEAN_IAT_S
                         MEAN_IAT_B       MEAN_DEPTH
                         MEAN_TD          DEV_TD
                         MEAN_SD          DEV_SD
                         MEAN_BD          DEV_BD
  REAL                 : TIDE_AMPLITUDE_1 TIDE_AMPLITUDE_2
                         TIDE_PERIOD_1    TIDE_PERIOD_2
                         TIDE_ALPHA_1     TIDE_ALPHA_2
  REAL                 : SIMULATIONTIME   RAND
                         MAINTENANCE_TIME MAINTENANCE_DURATION
                         TAB_X            TAB_Y
  INTEGER              : H I J L M N      PORT_CAP
                         DAYNUMBER        SATSUN
                         NR_CLASSES       NR_TERMINALS
                         NR_STRIKES       NR_TYPHOONS
```

```
                                LENGTH_ARRIVALPATT    LENGTH_DEPARTUREPATT
                                ERRORS
   LOGICAL                    : RESTR_STRIKE          RESTR_TYPHOON
                                SPECIFIED             MOORING[10]
   REFERENCE TO SET           : CRANESET              TERMINALSET
                                CLASSES
   CONTINUOUS(0)              : ENTRANCE_DEPTH
   TABLE(20)                  : TAB_DRAUGHT[4]        TAB_LENGTH[4]

 ATTRIBUTES OF GENERATOR:
   REAL                       : INTERARRIVALTIME
   REFERENCE TO SHIPCLASS     : MYCLASS
   REFERENCE TO SHIP          : NEXTSHIP

 ATTRIBUTES OF SHIPCLASS:
   REAL                       : MEAN_I    DEV_I       LOW_I    UP_I
                                MEAN_E    DEV_E       LOW_E    UP_E
                                MEAN_M    DEV_M       LOW_M
                                MEAN_C    DEV_C       LOW_DWT  UP_DWT
                                IAT       DEV_IAT
   REAL                       : CLASS_WORK_TIME       CLASS_WAITINGTIME
                                CLASS_BERTHTIME       CLASS_MOORINGTIME
                                CLASS_GEAR_LOADCAP    CLASS_GEAR_UNLOADCAP
   REAL                       : CLASSWAIT_TYPHOON     CLASSWAIT_TIDE
                                CLASSDELAY_TIDE       CLASSDELAY_STRIKE
                                CLASSDELAY_TYPHOON    CLASSDELAY_BREAKDOWN
                                CLASSDELAY_SHIFTING   CLASSDELAY_LEAVING
   INTEGER                    : CLASS_NUMBER          CLASS_CARGOTYPE[6]
                                CLASS_TERM_PERC       CLASS_ALLSHIPS
                                CLASS_CRANE_DEMAND    CLASS_FEEDER_PERC
                                CLASS_TERM_TAB[6]     CLASS_CODE
   LOGICAL                    : CLASS_FEEDER          CLASS_PRIORITY

 ATTRIBUTES OF SHIP:
   REAL                       : SHIP_IMPORTTOTAL      SHIP_EXPORTTOTAL
                                SHIP_LENGTH_BERTHED   SHIP_DRAUGHT_BERTHED
                                SHIPCALL1             SHIPCALL2
                                SHIPCALL3             SHIP_RATE
   REAL                       : SHIP_LOAD_RATE        SHIP_UNLOAD_RATE
                                SHIP_COVER_FAC        SHIP_FORTYFEETFAC_EXP
                                SHIP_MOORINGTIME      SHIP_FORTYFEETFAC_IMP
                                SHIP_X_BOW            SHIP_X_STERN
                                SHIP_DWT              SHIP_MOORING_EXTRA
   INTEGER                    : SHIP_PERC_ARR[24]     SHIP_PERC_DEP[24]
```

```
                                SHIP_CARGOTYPE       SHIP_TERM_NUMBER
                                SHIP_EXTRA_ARR       SHIP_EXTRA_DEP
                                SHIP_CRANE_SUPPLY    SHIP_DAY
  LOGICAL                     : SHIP_LOADING         SHIP_UNLOADING
                                SHIP_SHIFTED         SHIP_CRANE_ALLOC
  CONTINUOUS(1)               : SHIP_LOAD
  REFERENCE TO CRANE          : RIGHTCRANE
  REFERENCE TO BERTH          : RIGHTBERTH
  REFERENCE TO TERMINAL       : RIGHTTERM
  REFERENCE TO SHIPCLASS      : RIGHTCLASS

ATTRIBUTES OF HARBOUR_MASTER:
  INTEGER                     : HM_ROWLENGTH         HM_PORTLENGTH
  REFERENCE TO SHIP           : HM_CHECKSHIP         HM_SHIFTSHIP
                                HM_PRIORITY_SHIP
  REFERENCE TO BERTH          : HM_CHECKBERTH

ATTRIBUTES OF TERMINAL_MASTER:
  REAL                        : TM_LOAD_A            TM_LOAD_B
                                TM_LOAD_C
  INTEGER                     : TM_QUAYLENGTH
  REFERENCE TO SHIP           : TM_SHIP_A            TM_SHIP_B
                                TM_SHIP_C

ATTRIBUTES OF TERMINAL:
  REAL                        : TERM_ARR_TRUCK       TERM_DEP_TRUCK
                                TERM_ARR_BARGE       TERM_DEP_BARGE
                                TERM_ARR_WAGON       TERM_DEP_WAGON
  REAL                        : TERM_OPERATION_FAC   TERM_NET_FAC
                                TERM_VOL_OCCUPIED    TERM_VOL_CAPACITY
                                TERM_AREA_OCCUPIED   TERM_AREA_CAPACITY
                                TERM_EQUIP_CAP       TERM_EQUIP_PERC
  INTEGER                     : TERMINAL_NUMBER      TERM_SHIFTS
                                TERM_BREAKDOWNS      TERM_CARGOTYPES
                                TERM_BERTHS          TERM_CRANES
                                TERM_EQUIP_UNITS     TERM_ALLSHIPS
  LOGICAL                     : TERM_SAT_TRANS       TERM_SUN_TRANS
                                TERM_SAT_LOAD        TERM_SUN_LOAD
                                TERM_SHIFT
  REFERENCE TO SET            : TERM_CARGO           TERM_SHIPS
                                TERM_BERTHSET

ATTRIBUTES OF SHIFT:
  REAL                        : SHIFT_TIME[3]        SHIFT_DURATION[3]
  INTEGER                 : K
  REFERENCE TO TERMINAL   : SHIFT_TERM

ATTRIBUTES OF CARGOTYPE:
  REAL                    : CT_EXP_ARR_COV        CT_EXP_ARR_OPEN
                            CT_IMP_DEP_COV        CT_IMP_DEP_OPEN
                            CT_THROUGHPUT_EXP     CT_THROUGHPUT_IMP
                            CT_STACKHEIGHT_EXP    CT_STACKHEIGHT_IMP
                            CT_STACKHEIGHT_EMP    CT_STACKHEIGHT_GEN_COV
                                                  CT_STACKHEIGHT_GEN_OPEN
  REAL                    : CT_STORAGE_EXP_COV    CT_STORAGE_IMP_COV
                            CT_STORAGE_EXP_OPEN   CT_STORAGE_IMP_OPEN
                            CT_STORAGE_FEEDER     CT_STORAGE_VOLUME
                            CT_STORAGE_AREA_COV   CT_STORAGE_AREA_OPEN
                            CT_DENSITY            CT_GROSS_FAC
  REAL                    : CT_STACK_EMP_EXP      CT_STACK_EMP_IMP
                            CT_STACK_EXP          CT_STACK_IMP
                            CT_GRSLOT_EXP         CT_CAP_TRUCK
                            CT_GRSLOT_IMP         CT_CAP_WAGON
                            CT_GRSLOT_EMP         CT_CAP_BARGE
  INTEGER                 : CT_PERC_ARR_ROAD      CT_PERC_DEP_ROAD
                            CT_PERC_ARR_RAIL      CT_PERC_DEP_RAIL
                            CT_PERC_ARR_IWT       CT_PERC_DEP_IWT
                            CT_PERC_ARR_DIRECT    CT_PERC_DEP_DIRECT
  INTEGER                 : CT_EMPTIESPERC_EXP    CT_EMPTIESPERC_IMP
                            CT_FORTYFEETPERC_EXP CT_FORTYFEETPERC_IMP
                            CT_NUMBER
  CHARACTER(1)            : CT_CODE
  REFERENCE TO SET        : CT_ARRIVALPATT        CT_DEPARTUREPATT

ATTRIBUTES OF CRANE:
  REAL                    : CRANE_BREAKDOWN_TIME CRANE_BREAKDOWN_DURATION
                            CRANE_RESTTIME        CRANE_DELAYTIME
                            CRANE_LOADCAP         CRANE_UNLOADCAP
  REAL                    : CRANE_WORKTIME_A      CRANE_WORKTIME_B
                            CRANE_DOWNTIME        CRANECALL
                            CRANE_X_MIN           CRANE_X_MAX
  INTEGER                 : CRANE_BERTHNUMBER     CRANE_NUMBER
  LOGICAL                 : CRANE_AVAILABILITY
  REFERENCE TO SHIP       : CRANE_MYSHIP          CRANE_PREV_SHIP

ATTRIBUTES OF BERTH:
  REAL                    : BERTHWAIT_TYPHOON     BERTHWAIT_TIDE
                            BERTHDELAY_STRIKE     BERTHDELAY_SHIFTING
```

```
                              BERTHDELAY_BREAKDOWN BERTHDELAY_TYPHOON
                              BERTHDELAY_LEAVING
  REAL                      : BERTH_OCC_TIME       BERTH_WORK_TIME
                              BERTH_LENGTH         BERTH_DEPTH
                              BERTH_LENGTH_FREE    BERTH_SPACE_FREE
  INTEGER                   : BERTH_TERM_NUMBER    BERTH_NUMBER
                              BERTH_CARGOTYPES     BERTH_CARGOTYPE[6]
                              BERTH_CRANE_SHIFTS   BERTH_SHIP_SHIFTS
                              BERTH_CRANE_SUPPLY   BERTH_CAP
  LOGICAL                   : BERTH_SINGLE
  REFERENCE TO SET          : BERTH_CRANES         BERTH_SHIPS

ATTRIBUTES OF ARRIVALDAY:
  INTEGER                   : ARRDAY_PERC          ARRDAY_NUM

ATTRIBUTES OF DEPARTUREDAY:
  INTEGER                   : DEPDAY_PERC          DEPDAY_NUM

ATTRIBUTES OF TYPHOON:
  REAL                      : TYPHOON_CALL         TYPHOON_DURATION
                              TYPHOON_INTERARRIVALTIME
  INTEGER                   : TYPHOONBEGIN         TYPHOONEND
                              TYPHOON_SPELL
  LOGICAL                   : TYPHOON_ALARM

ATTRIBUTES OF STRIKE:
  REAL                      : STRIKE_CALL          STRIKE_DURATION
                              STRIKE_INTERARRIVALTIME
  INTEGER                   : STRIKE_SPELL
  LOGICAL                   : STRIKE_ALARM

```

```
MODEL HP2_11P                                                       Date:  93/08/16
MOD MAINMOD                                                         Time:  18:27:47
```

```
@@@@@@@@@@@@@@@@@
@  MAIN MODULE  @
@@@@@@@@@@@@@@@@@

CRANESET    ← NEW SET
TERMINALSET ← NEW SET
CLASSES     ← NEW SET
REWIND CHECKLIST
WRITE "############################" TO CHECKLIST WITH IMAGE A
WRITE "CHECKLIST FOR:" TO CHECKLIST WITH IMAGE A
WRITE "- ERRORS IN INPUT-FILES" TO CHECKLIST WITH IMAGE A
WRITE "- CORRECTNESS OF REFERENCES" TO CHECKLIST WITH IMAGE A
WRITE "############################" TO CHECKLIST WITH IMAGE A
WRITE " " TO CHECKLIST WITH IMAGE A
WRITE " " TO CHECKLIST WITH IMAGE A
WRITE "----------------" TO CHECKLIST WITH IMAGE A
WRITE "REPORT OF ERRORS" TO CHECKLIST WITH IMAGE A
WRITE "----------------" TO CHECKLIST WITH IMAGE A
WRITE " " TO CHECKLIST WITH IMAGE A

@ ---------------------------- @
@  Create harbour-environment  @
@ ---------------------------- @

TYPHOONBEGIN ← READ FROM HFILE
CALL ERROR1("TYP_BEGIN",4) IF TYPHOONBEGIN > 365
TYPHOONEND ← READ FROM HFILE
CALL ERROR1("TYP_END",4) IF TYPHOONEND > 365
MEAN_IAT_T ← READ FROM HFILE
CALL ERROR1("MEAN_IAT_T",1) IF MEAN_IAT_T = 0
MEAN_TD    ← READ FROM HFILE
DEV_TD     ← READ FROM HFILE
MEAN_IAT_S ← READ FROM HFILE
CALL ERROR1("MEAN_IAT_S",1) IF MEAN_IAT_S = 0
MEAN_SD ← READ FROM HFILE
DEV_SD  ← READ FROM HFILE
RESHAPE EXP_IAT_T AS SAMPLED FROM DISTRIBUTION EXPONENTIAL WITH PARAMETER
MEAN(MEAN_IAT_T)
RESHAPE NORM_TD AS SAMPLED FROM DISTRIBUTION NORMAL WITH PARAMETERS
MEAN(MEAN_TD) DEVIATION(DEV_TD)
RESHAPE EXP_IAT_S AS SAMPLED FROM DISTRIBUTION EXPONENTIAL WITH PARAMETER
```

```
   MEAN(MEAN_IAT_S)
40 RESHAPE NORM_SD AS SAMPLED FROM DISTRIBUTION NORMAL WITH PARAMETERS
   MEAN(MEAN_SD) DEVIATION(DEV_SD)
41 SEED OF EXP_IAT_T ← READ FROM HFILE
42 SEED OF NORM_TD   ← READ FROM HFILE
43 SEED OF EXP_IAT_S ← READ FROM HFILE
44 SEED OF NORM_SD   ← READ FROM HFILE
45 MEAN_DEPTH        ← READ FROM HFILE
46 TIDE_AMPLITUDE_1  ← READ FROM HFILE
47 TIDE_PERIOD_1     ← READ FROM HFILE
48 TIDE_ALPHA_1      ← READ FROM HFILE
49 TIDE_AMPLITUDE_2  ← READ FROM HFILE
50 TIDE_PERIOD_2     ← READ FROM HFILE
51 TIDE_ALPHA_2      ← READ FROM HFILE
52 L ← READ FROM HFILE
53 RESTR_STRIKE ← 1 = L
54 CALL ERROR1("RESTR_STR",2) IF (L = 0) ^ (L = 1)
55 L ← READ FROM HFILE
56 RESTR_TYPHOON ← 1 = L
57 CALL ERROR1("RESTR_TYP",2) IF (L = 0) ^ (L = 1)
58 SIMULATIONTIME ← READ FROM HFILE
59 CALL ERROR3("H_FILE") IF READ FROM HFILE = ~1
60 SPECIFY ENTRANCE_DEPTH PRECEPT(ENTRANCE_DEPTH ← MEAN_DEPTH +
   (TIDE_AMPLITUDE_1 x COS((CTx(2x3.141593)÷TIDE_PERIOD_1) - TIDE_ALPHA_1))+
   (TIDE_AMPLITUDE_2 x COS((CTx(2x3.141593)÷TIDE_PERIOD_2) - TIDE_ALPHA_2)))
61 DISPLAY "HARBOUR-ENVIRONMENT HAS BEEN CREATED" AT LINE 1 POSITION 1 WITH
   IMAGE A
62
63 @ ----------------------------- @
64 @  Create terminal(s)            @
65 @ ----------------------------- @
66
67 NR_TERMINALS           ← READ FROM TFILE
68 LENGTH_ARRIVALPATT     ← READ FROM TFILE
69 LENGTH_DEPARTUREPATT   ← READ FROM TFILE
70 I ← 0
71 SATSUN ← 2
72 WHILE ((I x 5) + 5) < LENGTH_ARRIVALPATT
73    SATSUN ← SATSUN + 2
74    I ← I + 1
75 END
76 LENGTH_ARRIVALPATT ← LENGTH_ARRIVALPATT + SATSUN
77
78 FOR H ← 1 TO NR_TERMINALS
```

```
THIS TERMINAL   ← NEW TERMINAL
THIS SHIFT      ← NEW SHIFT
TERM_CARGO      ← NEW SET
TERM_SHIPS      ← NEW SET
TERM_BERTHSET   ← NEW SET
TERMINAL_NUMBER ← H

@  Create cargo-types  @
TERM_CARGOTYPES  ← READ FROM TFILE
FOR I ← 1 TO TERM_CARGOTYPES
   THIS CARGOTYPE      ← NEW CARGOTYPE CALLED CHREAD FROM TFILE
   CT_NUMBER           ← READ FROM TFILE
   CT_CODE             ← CHREAD FROM TFILE
   CT_ARRIVALPATT      ← NEW SET
   CT_DEPARTUREPATT    ← NEW SET
   CALL PATTERNS
   IF CT_CODE = "A"
      CT_STACKHEIGHT_IMP   ← READ FROM TFILE
      CT_STACKHEIGHT_EXP   ← READ FROM TFILE
      CT_STACKHEIGHT_EMP   ← READ FROM TFILE
      CT_EMPTIESPERC_IMP   ← READ FROM TFILE
      CT_EMPTIESPERC_EXP   ← READ FROM TFILE
      CT_FORTYFEETPERC_IMP ← READ FROM TFILE
      CT_FORTYFEETPERC_EXP ← READ FROM TFILE
   END
   IF CT_CODE = "B"
      CT_STACKHEIGHT_GEN_COV  ← READ FROM TFILE
      CT_STACKHEIGHT_GEN_OPEN ← READ FROM TFILE
      CT_DENSITY              ← READ FROM TFILE
   END
   CT_DENSITY ← READ FROM TFILE
   IF (CT_CODE = "C") v (CT_CODE = "D") v (CT_CODE = "E")

   CT_GROSS_FAC ← READ FROM TFILE
   CT_CAP_TRUCK ← READ FROM TFILE
   CT_CAP_WAGON ← READ FROM TFILE
   CT_CAP_BARGE ← READ FROM TFILE
   JOIN THIS CARGOTYPE TO TERM_CARGO
   L ← READ FROM TFILE
   IF L = ~1
      WRITE "FATAL ERROR:" TO CHECKLIST WITH IMAGE A
      WRITE "DATA OVERFLOW FOR CHARACTERISTICS OF CARGO-TYPE";
      CT_NUMBER TO CHECKLIST WITH IMAGE A^xxx
      WRITE " " TO CHECKLIST WITH IMAGE A
```

```
            DISPLAY "FATAL ERROR IN T-FILE, CAUSING IMMEDIATE INTERRUPTION"
            AT LINE 2 POSITION 5 WITH IMAGE A
            DISPLAY "OF DATA-READING. ERROR CONCERNS INPUT-DATA FOR"
            AT LINE 3 POSITION 5 WITH IMAGE A
            DISPLAY "CARGO-TYPE";CT_NUMBER AT LINE 4 POSITION 5 WITH
            IMAGE A^xxx
            ERRORS ← ERRORS + 1
         END
         INTERRUPT IF L = ~1
      END

      @  Create berths  @
      TERM_BERTHS ← READ FROM TFILE
      FOR I ← 1 TO TERM_BERTHS
         THIS BERTH          ← NEW BERTH
         BERTH_TERM_NUMBER   ← H
         BERTH_NUMBER        ← READ FROM TFILE
         L ← READ FROM TFILE
         BERTH_SINGLE ← 1 = L
         CALL ERROR1("BER_SING  "|BERTH_NUMBER,2) IF (L = 0) ^ (L = 1)
         BERTH_LENGTH        ← READ FROM TFILE
         BERTH_DEPTH         ← READ FROM TFILE
         BERTH_CRANE_SUPPLY  ← READ FROM TFILE
         BERTH_CARGOTYPES    ← READ FROM TFILE
         BERTH_LENGTH_FREE   ← BERTH_LENGTH
         BERTH_CRANES        ← NEW SET
         BERTH_SHIPS         ← NEW AUTOSTORING SET WITH STATISTICS
         CALLED "Q_BERTH  "|BERTH_NUMBER
         FOR J ← 1 TO BERTH_CARGOTYPES
            BERTH_CARGOTYPE[J] ← READ FROM TFILE
         END
         BERTH_CAP ← READ FROM TFILE
         CALL ERROR1("BER_CAP  "|BERTH_NUMBER,4) IF (BERTH_SINGLE = TRUE) ^
         (BERTH_CAP > 1)
         PORT_CAP  ← PORT_CAP + BERTH_CAP
         JOIN THIS BERTH TO TERM_BERTHSET RANKED BY BERTH_DEPTH
      END

      @  Create cranes  @
      TERM_CRANES ← READ FROM TFILE
      FOR I ← 1 TO TERM_CRANES
         THIS CRANE ← NEW CRANE CALLED CHREAD FROM TFILE
         CRANE_NUMBER ← READ FROM TFILE
         CRANE_X_MIN  ← READ FROM TFILE
      CRANE_X_MAX  ← READ FROM TFILE
      CRANE_LOADCAP     ← READ FROM TFILE
      CRANE_UNLOADCAP   ← READ FROM TFILE
      CRANE_BERTHNUMBER ← READ FROM TFILE
      THIS BERTH ← FIRST BERTH IN TERM_BERTHSET WITH BERTH_NUMBER =
      CRANE_BERTHNUMBER
      CALL ERROR1("XMIN_CR  "¦CRANE_NUMBER,4) IF CRANE_X_MIN >
      BERTH_LENGTH
      CALL ERROR1("XMAX_CR  "¦CRANE_NUMBER,4) IF CRANE_X_MAX >
      BERTH_LENGTH
      JOIN THIS CRANE TO CRANESET
      JOIN THIS CRANE TO BERTH_CRANES RANKED BY
      1 ÷ CRANE_LOADCAP + CRANE_UNLOADCAP IF BERTH_SINGLE = TRUE
      JOIN THIS CRANE TO BERTH_CRANES RANKED BY
      CRANE_NUMBER IF BERTH_SINGLE = FALSE
   END

   @  Additional information for this terminal  @
   L ← READ FROM TFILE
   TERM_SAT_TRANS ← 1 = L
   CALL ERROR1("SAT_TR_ "¦H,2) IF (L ■ 0) ^ (L ■ 1)
   L ← READ FROM TFILE
   TERM_SUN_TRANS ← 1 = L
   CALL ERROR1("SAT_LO_ "¦H,2) IF (L ■ 0) ^ (L ■ 1)
   L ← READ FROM TFILE
   TERM_SAT_LOAD ← 1 = L
   CALL ERROR1("SUN_TR_ "¦H,2) IF (L ■ 0) ^ (L ■ 1)
   L ← READ FROM TFILE
   TERM_SUN_LOAD ← 1 = L
   CALL ERROR1("SUN_LO_" ¦H,2) IF (L ■ 0) ^ (L ■ 1)
   TERM_AREA_CAPACITY ← READ FROM TFILE
   TERM_VOL_CAPACITY  ← READ FROM TFILE
   TERM_EQUIP_UNITS   ← READ FROM TFILE
   TERM_EQUIP_CAP     ← READ FROM TFILE
   TERM_EQUIP_PERC    ← READ FROM TFILE
   TERM_OPERATION_FAC ← READ FROM TFILE
   TERM_SHIFTS        ← READ FROM TFILE
   SHIFT_TERM         ← THIS TERMINAL
   FOR I ← 1 TO TERM_SHIFTS
      SHIFT_TIME[I]     ← READ FROM TFILE
      SHIFT_DURATION[I] ← READ FROM TFILE
      TERM_NET_FAC ← TERM_NET_FAC + SHIFT_DURATION[I] ÷ 24
   END
   CALL ERROR1("SHIFTTERM "¦H,4) IF TERM_NET_FAC > 1
   JOIN THIS TERMINAL TO TERMINALSET
   ACTIVATE THIS SHIFT FROM SHIFTSTART IN SHIFTMOD
   DISPLAY "TERMINAL";H;"HAS BEEN CREATED" AT LINE 1+H POSITION 1 WITH
   IMAGE A^xx^A
END

@  General information for all terminals  @
MEAN_IAT_B   ← READ FROM TFILE ÷ 24
CALL ERROR1("MEAN_IAT_B",1) IF MEAN_IAT_B = 0
MEAN_BD                ← READ FROM TFILE
DEV_BD                 ← READ FROM TFILE
MAINTENANCE_TIME       ← READ FROM TFILE
MAINTENANCE_DURATION ← READ FROM TFILE
RESHAPE EXP_IAT_B AS SAMPLED FROM DISTRIBUTION EXPONENTIAL WITH PARAMETER
MEAN(MEAN_IAT_B)
RESHAPE NORM_BD AS SAMPLED FROM DISTRIBUTION NORMAL WITH PARAMETERS
MEAN(MEAN_BD) DEVIATION (DEV_BD)
SEED OF EXP_IAT_B ← READ FROM TFILE
SEED OF NORM_BD   ← READ FROM TFILE
FOR EACH CRANE IN CRANESET
   THIS TERMINAL ← FIRST TERMINAL IN TERMINALSET WITH TERMINAL_NUMBER =
   BERTH_TERM_NUMBER
   CRANE_LOADCAP              ← CRANE_LOADCAP x 24 x TERM_OPERATION_FAC
   CRANE_UNLOADCAP            ← CRANE_UNLOADCAP x 24 x TERM_OPERATION_FAC
   CRANE_AVAILABILITY         ← TRUE
   CRANE_BREAKDOWN_TIME       ← 10000000    @ EXP_IAT_B
   CRANE_BREAKDOWN_DURATION ← NORM_BD
   CRANECALL                  ← NOW
END
RESHAPE UNIF AS SAMPLED FROM DISTRIBUTION UNIFORM
SEED OF UNIF ← READ FROM TFILE
CALL ERROR3("T_FILE") IF READ FROM TFILE ■ ~1

@ -------------------- @
@  Create ship-classes  @
@ -------------------- @

NR_CLASSES ← READ FROM SFILE
FOR I ← 1 TO NR_CLASSES
   THIS SHIPCLASS ← NEW SHIPCLASS CALLED CHREAD FROM SFILE
   L ← READ FROM SFILE
   CLASS_FEEDER ← 1 = L
   CALL ERROR1("CL_FEED "¦I,2) IF (L ■ 0) ^ (L ■ 1)
   IAT ← READ FROM SFILE
```

```
239    CALL ERROR1("IAT-CLASS "¦I,1) IF IAT = 0
240    DEV_IAT ← READ FROM SFILE
241    CLASS_FEEDER_PERC ← READ FROM SFILE
242    MEAN_I ← READ FROM SFILE
243    DEV_I  ← READ FROM SFILE
244    LOW_I  ← READ FROM SFILE
245    CALL ERROR1("MINIMP_CL "¦I,4) IF LOW_I > MEAN_I
246    UP_I ← READ FROM SFILE
247    CALL ERROR1("MAXIMP_CL "¦I,3) IF UP_I < MEAN_I
248    MEAN_E ← READ FROM SFILE
249    DEV_E  ← READ FROM SFILE
250    LOW_E  ← READ FROM SFILE
251    CALL ERROR1("MINEXP_CL "¦I,4) IF LOW_E > MEAN_E
252    UP_E ← READ FROM SFILE
253    CALL ERROR1("MAXEXP_CL "¦I,3) IF UP_E < MEAN_E
254    LOW_DWT ← READ FROM SFILE
255    UP_DWT  ← READ FROM SFILE
256    CALL ERROR1("MAXDWT_CL "¦I,3) IF UP_DWT < LOW_DWT
257    MEAN_M ← READ FROM SFILE
258    DEV_M  ← READ FROM SFILE
259    LOW_M  ← READ FROM SFILE
260    CALL ERROR1("MINMOO_CL "¦I,4) IF LOW_M > MEAN_M
261    MEAN_C ← READ FROM SFILE
262    DEV_C  ← READ FROM SFILE
263    RESHAPE EXP_IAT[I] AS SAMPLED FROM DISTRIBUTION EXPONENTIAL WITH
       PARAMETER MEAN(IAT) IF CLASS_FEEDER = FALSE
264    RESHAPE NORM_IAT[I] AS SAMPLED FROM DISTRIBUTION NORMAL WITH PARAMETER
       MEAN(IAT) DEVIATION(DEV_IAT) IF CLASS_FEEDER = TRUE
265    RESHAPE NORM_I[I] AS SAMPLED FROM DISTRIBUTION EXPONENTIAL WITH
       PARAMETERS MEAN(MEAN_I)
266    RESHAPE NORM_E[I] AS SAMPLED FROM DISTRIBUTION EXPONENTIAL WITH
       PARAMETERS MEAN(MEAN_E)
267    RESHAPE UNIF_DWT[I] AS SAMPLED FROM DISTRIBUTION UNIFORM WITH
       PARAMETERS LB(LOW_DWT) UB(UP_DWT)
268    RESHAPE UNKN_M[I] AS SAMPLED FROM DISTRIBUTION UNKNOWN WITH PARAMETERS
       MEAN(MEAN_M) DEVIATION(DEV_M) LB(LOW_M)
269    RESHAPE NORM_C[I] AS SAMPLED FROM DISTRIBUTION NORMAL WITH PARAMETERS
       MEAN(MEAN_C) DEVIATION(DEV_C)
270    SEED OF NORM_IAT[I]  ← 2910 + I x 9326
271    SEED OF EXP_IAT[I]   ← 4820 + I x 1133
272    SEED OF NORM_I[I]    ← 3915 + I x 1748
273    SEED OF NORM_E[I]    ← 8743 + I x 1091
274    SEED OF UNIF_DWT[I]  ← 9562 + I x 2030
275    SEED OF UNKN_M[I]    ← 7276 + I x 3470
```

```
   SEED OF NORM_C[I]     ← 3353 + I
   CLASS_NUMBER          ← I
   CLASS_CRANE_DEMAND    ← READ FROM SFILE
   CLASS_GEAR_LOADCAP    ← READ FROM SFILE
   CLASS_GEAR_UNLOADCAP  ← READ FROM SFILE
   L                     ← READ FROM SFILE
   CLASS_PRIORITY        ← 1 = L
   CALL ERROR1("CL_PRIOR "¦I,2) IF (L ≠ 0) ^ (L ≠ 1)
   CLASS_CODE            ← READ FROM SFILE
   CALL ERROR1("CL_CODE "¦I,2) IF (CLASS_CODE ≠ 1) ^ (CLASS_CODE ≠ 2) ^
   (CLASS_CODE ≠ 3) ^ (CLASS_CODE ≠ 4)
   FOR H ← 1 TO NR_TERMINALS
      CLASS_TERM_PERC    ← CLASS_TERM_PERC + READ FROM SFILE
      CLASS_TERM_TAB[H]  ← CLASS_TERM_PERC
      CLASS_CARGOTYPE[H] ← READ FROM SFILE
   END
   CALL ERROR2("CLASSTERM-",I) IF CLASS_TERM_PERC ≠ 100
   JOIN THIS SHIPCLASS TO CLASSES
   THIS GENERATOR ← NEW GENERATOR
   MYCLASS ← THIS SHIPCLASS
   ACTIVATE THIS GENERATOR FROM GEN_START IN GENERATORMOD
   DISPLAY "SHIPCLASS";I;"HAS BEEN CREATED" AT LINE 1+NR_TERMINALS+I
   POSITION 1 WITH IMAGE A^xx^A
END
CALL ERROR3("S_FILE") IF READ FROM SFILE ≠ ~1
CALL DWT_TABLES
CALL ERROR3("D_FILE") IF READ FROM DFILE ≠ ~1

@ --------------------------- @
@  Checklist\Report of errors  @
@ --------------------------- @

FOR I ← 1 TO 20
   DISPLAY "                                                      " AT LINE I
   POSITION 1 WITH IMAGE A
END
DISPLAY "CHECK C-FILE FOR CORRECTNESS OF REFERENCE-NUMBERS" AT LINE 5
POSITION 5 WITH IMAGE A
DISPLAY "TOTAL NUMBER OF ERRORS IN INPUT-FILES IS";ERRORS AT LINE 7
POSITION 5 WITH IMAGE A^xxx
IF ERRORS > 0
   DISPLAY "LEAVE RUN-TIME MENU AND CHANGE INPUT-FILES" AT LINE 8
   POSITION 5 WITH IMAGE A
   DISPLAY "CHECK C-FILE FOR SPECIFICATION OF ERRORS" AT LINE 9
```

```
    POSITION 5 WITH IMAGE A
314 END
315 DISPLAY "CHECK STATE-ANALYSIS FOR CORRECTNESS OF OTHER INPUT-DATA" AT
    LINE 11 POSITION 5 WITH IMAGE A
316 WRITE " " TO CHECKLIST WITH IMAGE A
317 WRITE "##############################" TO CHECKLIST WITH IMAGE A
318 WRITE "TOTAL NUMBER OF ERRORS IS";ERRORS TO CHECKLIST WITH IMAGE A^xxx
319 WRITE "##############################" TO CHECKLIST WITH IMAGE A
320 WRITE " " TO CHECKLIST WITH IMAGE A
321 WRITE " " TO CHECKLIST WITH IMAGE A
322 WRITE "---------------------" TO CHECKLIST WITH IMAGE A
323 WRITE "CARGOTYPE-REFERENCES" TO CHECKLIST WITH IMAGE A
324 WRITE "---------------------" TO CHECKLIST WITH IMAGE A
325 WRITE " " TO CHECKLIST WITH IMAGE A
326 WRITE "ACCORDING TO S-FILE:" TO CHECKLIST WITH IMAGE A
327 WRITE "TERMINAL    " TO CHECKLIST WITH IMAGE A
328 FOR EACH TERMINAL IN TERMINALSET
329    WRITE TERMINAL_NUMBER;"    " TO CHECKLIST WITH IMAGE |xx^A
330 END
331 FOR EACH SHIPCLASS IN CLASSES
332    L ← 0
333    WRITE "CLASS";CLASS_NUMBER TO CHECKLIST WITH IMAGE A^xxx
334    FOR EACH TERMINAL IN TERMINALSET
335       L ← L + 1
336       WRITE "   ";CLASS_CARGOTYPE[L] TO CHECKLIST WITH IMAGE |A^xxx IF
          CLASS_CARGOTYPE[L] ≠ 0
337       WRITE "    -  "; TO CHECKLIST WITH IMAGE |A IF CLASS_CARGOTYPE[L]=0
338    END
339 END
340 WRITE " " TO CHECKLIST WITH IMAGE A
341 WRITE "ACCORDING TO T-FILE:" TO CHECKLIST WITH IMAGE A
342 FOR EACH TERMINAL IN TERMINALSET
343    WRITE "CARGOTYPES ON TERMINAL";TERMINAL_NUMBER;":" TO CHECKLIST WITH
       IMAGE A^xxA
344    FOR EACH CARGOTYPE IN TERM_CARGO
345       WRITE "   ";CT_NUMBER TO CHECKLIST WITH IMAGE |A^xxx
346    END
347    WRITE "CARGOTYPES AT BERTH:" TO CHECKLIST WITH IMAGE A
348    FOR EACH BERTH IN TERM_BERTHSET
349       WRITE BERTH_NUMBER;":" TO CHECKLIST WITH IMAGE xxA
350       FOR I ← 1 TO BERTH_CARGOTYPES
351          WRITE " ";BERTH_CARGOTYPE[I] TO CHECKLIST WITH IMAGE |Axxx
352       END
353    END
```

```
END
WRITE " " TO CHECKLIST WITH IMAGE A
WRITE " " TO CHECKLIST WITH IMAGE A
WRITE "---------------------" TO CHECKLIST WITH IMAGE A
WRITE "BERTH\CRANE-REFERENCES" TO CHECKLIST WITH IMAGE A
WRITE "---------------------" TO CHECKLIST WITH IMAGE A
WRITE " " TO CHECKLIST WITH IMAGE A
FOR EACH TERMINAL IN TERMINALSET
   FOR EACH BERTH IN TERM_BERTHSET
      WRITE "BERTH";BERTH_NUMBER;"   NUMBER OF CRANES";BERTH_CRANE_SUPPLY
      TO CHECKLIST WITH IMAGE A^xx^A^xx
      WRITE "ALLOCATED TO THIS BERTH:" TO CHECKLIST WITH IMAGE A
      FOR EACH CRANE IN BERTH_CRANES
         WRITE " ";NAME OF THIS CRANE TO CHECKLIST WITH IMAGE ¦A^A
      END
      WRITE " " TO CHECKLIST WITH IMAGE A
   END
END
WRITE " " TO CHECKLIST WITH IMAGE A
WRITE "#######################" TO CHECKLIST WITH IMAGE A
WRITE "CHECK ALL REFERENCES !!" TO CHECKLIST WITH IMAGE A
WRITE "#######################" TO CHECKLIST WITH IMAGE A
WRITE " " TO CHECKLIST WITH IMAGE A
INTERRUPT

@ ---------------- @
@  Run simulation  @
@ ---------------- @

DAYNUMBER ← 0
ACTIVATE STORAGE_MASTER FROM  SM_START IN STORAGEMOD
ACTIVATE HARBOUR_MASTER FROM  HM_START IN HARBOURMOD
ACTIVATE TERMINAL_MASTER FROM TM_START IN TERMINALMOD
ACTIVATE TYPHOON FROM TYPHOONSTART IN TYPHOONMOD IF RESTR_TYPHOON = TRUE
ACTIVATE STRIKE  FROM STRIKESTART  IN STRIKEMOD  IF RESTR_STRIKE = TRUE
INTEGRATE WHILE NOW < SIMULATIONTIME

@ ----- @
@  End  @
@ ----- @

FOR EACH CRANE IN CRANESET WITH CRANE_MYSHIP IS NONE
   CRANE_RESTTIME ← CRANE_RESTTIME + NOW - CRANECALL
END
```

```
FOR EACH CRANE IN CRANESET WITH (CRANE_MYSHIP IS NOT NONE) ^
    ((CRANE_WORKTIME_A + NOW - CRANECALL) < CRANE_BREAKDOWN_TIME)
   CRANE_WORKTIME_B ← CRANE_WORKTIME_B + SHIPCALL1 OF CRANE_MYSHIP -
    CRANECALL
   CRANE_RESTTIME ← CRANE_RESTTIME + NOW - SHIPCALL1 OF CRANE_MYSHIP
END
FOR EACH CRANE IN CRANESET WITH (CRANE_MYSHIP IS NOT NONE) ^
    ((CRANE_WORKTIME_A + NOW - CRANECALL) ≥ CRANE_BREAKDOWN_TIME)
   CRANE_DOWNTIME ← CRANE_DOWNTIME + SHIPCALL1 OF CRANE_MYSHIP - CRANECALL
   CRANE_RESTTIME ← CRANE_RESTTIME + NOW - SHIPCALL1 OF CRANE_MYSHIP
END
FOR EACH SHIP IN QUAY
   THIS BERTH       ← RIGHTBERTH
   THIS TERMINAL    ← RIGHTTERM
   THIS SHIPCLASS   ← RIGHTCLASS
   BERTH_OCC_TIME   ← BERTH_OCC_TIME + SHIPCALL1 - QUEUETIME OF THIS SHIP
   IN PORT + SHIP_MOORINGTIME ÷ 24 IF BERTH_SINGLE = TRUE
   BERTH_OCC_TIME_  ← BERTH_OCC_TIME + SHIP_LENGTH_BERTHED x (SHIPCALL1 -
    QUEUETIME_OF THIS SHIP IN PORT + SHIP_MOORINGTIME ÷ 24)
   IF BERTH_SINGLE = FALSE
   CLASS_BERTHTIME ← CLASS_BERTHTIME + SHIPCALL1 - QUEUETIME OF THIS SHIP
   IN PORT + SHIP_MOORINGTIME ÷ 24
   CLASS_ALLSHIPS_  ← CLASS_ALLSHIPS + 1
END
ACTIVATE REPORTER FROM REPORTSTART IN REPORTMOD
WAIT WHILE REPORTER IS ACTIVE
DISPLAY "THANK YOU FOR USING HASPORT-II" AT LINE 7 POSITION 5 WITH IMAGE A
CANCEL ALL
TERMINATE
```

MODEL HP2_11P   Date: 93/08/16
MOD GENERATORMOD   Time: 18:28:35

```
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@  PROCESS OF THE GENERATOR  @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

GEN_START:
  THIS SHIPCLASS ← MYCLASS
  INTERARRIVALTIME ← EXP_IAT[CLASS_NUMBER] IF CLASS_FEEDER = FALSE
  INTERARRIVALTIME ← NORM_IAT[CLASS_NUMBER] IF CLASS_FEEDER = TRUE
  WAIT INTERARRIVALTIME
  NEXTSHIP  ← NEW SHIP
  THIS SHIP ← NEXTSHIP

  @  Generate attributes of ship  @
  THIS SHIPCLASS        ← MYCLASS
  SHIP_IMPORTTOTAL      ← NORM_I[CLASS_NUMBER]
  SHIP_EXPORTTOTAL      ← NORM_E[CLASS_NUMBER]
  SHIP_DWT              ← UNIF_DWT[CLASS_NUMBER]
  SHIP_DRAUGHT_BERTHED ← (VALUE OF TAB_DRAUGHT[CLASS_CODE] AT (SHIP_DWT))
                          @ x 1.15
  SHIP_LENGTH_BERTHED  ← (VALUE OF TAB_LENGTH[CLASS_CODE] AT (SHIP_DWT))
                          @ x 1.10
  SHIP_MOORINGTIME      ← UNKN_M[CLASS_NUMBER]
  SHIP_COVER_FAC        ← NORM_C[CLASS_NUMBER]
  WHILE (SHIP_COVER_FAC < MAX(0,MEAN_C - 2xDEV_C)) v (SHIP_COVER_FAC >
     MIN(100,MEAN_C + 2 x DEV_C))
     SHIP_COVER_FAC     ← NORM_C[CLASS_NUMBER]
  END
  RIGHTCLASS            ← MYCLASS
  @  Allocate ship to a terminal  @
  RAND ← 100 x UNIF
  FOR H ← 1 TO NR_TERMINALS
     SHIP_TERM_NUMBER ← (NR_TERMINALS + 1) - H
     IF RAND ≤ CLASS_TERM_TAB[(NR_TERMINALS + 1) - H]
  END
  RIGHTTERM ← FIRST TERMINAL IN TERMINALSET WITH TERMINAL_NUMBER =
  SHIP_TERM_NUMBER
  SHIP_CARGOTYPE ← CLASS_CARGOTYPE[SHIP_TERM_NUMBER]
  THIS CARGOTYPE ← FIRST CARGOTYPE IN TERM_CARGO OF RIGHTTERM WITH
  CT_NUMBER = SHIP_CARGOTYPE
  SHIP_FORTYFEETFAC_EXP ← 1 + CT_FORTYFEETPERC_EXP ÷ 100 IF CT_CODE = "A"
  SHIP_FORTYFEETFAC_IMP ← 1 + CT_FORTYFEETPERC_IMP ÷ 100 IF CT_CODE = "A"
```

```
SHIP_FORTYFEETFAC_EXP ← 1 IF CT_CODE = "A"
SHIP_FORTYFEETFAC_IMP ← 1 IF CT_CODE = "A"

IF SPECIFIED = FALSE
   SPECIFY SHIP_LOAD PRECEPT(SHIP_LOAD′ ← SHIP_RATE)
   SPECIFIED ← TRUE
END
ACTIVATE NEXTSHIP FROM SHIPSTART IN SHIPMOD IF CT_CODE = "E"
ACTIVATE NEXTSHIP FROM PASSENGERSHIPSTART IN PASSENGERSHIPMOD
IF CT_CODE = "E"  @  for pontianak
REPEAT FROM GEN_START
```

```
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@  PROCESS OF THE HARBOUR-MASTER  @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

HM_START:
  WAIT WHILE ROW IS EMPTY
  WAIT WHILE LENGTH OF PORT = PORT_CAP
  HM_CHECKSHIP ← FIRST SHIP IN ROW
  FOR L ← 1 TO LENGTH OF ROW
    THIS SHIP      ← HM_CHECKSHIP
    THIS SHIPCLASS ← RIGHTCLASS
    THIS TERMINAL  ← RIGHTTERM
    IF RIGHTBERTH IS NONE
      HM_CHECKBERTH ← FIRST BERTH IN TERM_BERTHSET
      FOR M ← 1 TO LENGTH OF TERM_BERTHSET
        THIS BERTH ← HM_CHECKBERTH
        HM_PRIORITY_SHIP ← FIRST SHIP IN PRIORITY_ROW WITH
        (RIGHTBERTH IS NONE) ^ (SHIP_TERM_NUMBER = BERTH_TERM_NUMBER) ^
        (CLASS_PRIORITY OF RIGHTCLASS = TRUE) ^
        (SHIP_DRAUGHT_BERTHED ≤ BERTH_DEPTH)
        IF (BERTH_DEPTH ≥ SHIP_DRAUGHT_BERTHED) ^
          (BERTH_LENGTH_FREE ≥ SHIP_LENGTH_BERTHED) ^
          (LENGTH OF BERTH_SHIPS < BERTH_CAP)
          IF HM_PRIORITY_SHIP IS NOT NONE
            FOR N ← 1 TO BERTH_CARGOTYPES
              GOTO HM_CONTINUE IF (HM_PRIORITY_SHIP IS NOT THIS SHIP) ^
              (BERTH_CARGOTYPE[N] = SHIP_CARGOTYPE OF HM_PRIORITY_SHIP)
            END
          END
          FOR N ← 1 TO BERTH_CARGOTYPES
            IF BERTH_CARGOTYPE[N] = SHIP_CARGOTYPE
              JOIN THIS SHIP TO BERTH_SHIPS RANKED BY SHIP_X_BOW
              BERTH_LENGTH_FREE ← BERTH_LENGTH_FREE - SHIP_LENGTH_BERTHED
              RIGHTBERTH ← THIS BERTH
              REACTIVATE THIS SHIP
              REPEAT FROM HM_START
            END
          END
        END
        HM_CONTINUE:
        HM_PRIORITY_SHIP ← NONE
        HM_CHECKBERTH ← SUCC OF THIS BERTH IN TERM_BERTHSET
      END
    END
    HM_CHECKSHIP ← SUCC OF THIS SHIP IN ROW
  END
  HM_ROWLENGTH  ← LENGTH OF ROW
  HM_PORTLENGTH ← LENGTH OF PORT
  WAIT WHILE (LENGTH OF ROW=HM_ROWLENGTH) ^ (LENGTH OF PORT=HM_PORTLENGTH)
  REPEAT FROM HM_START
```

```
MODEL HP2_11P                                                  Date:  93/08/16
MOD TERMINALMOD                                                Time:  18:28:53

@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@  PROCESS OF THE TERMINAL-MASTER  @
@  for pontianak                   @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

TM_START:
  TM_QUAYLENGTH ← LENGTH OF QUAY
  WAIT WHILE LENGTH OF QUAY = TM_QUAYLENGTH
  GOTO TM_REALLOCATE IF LENGTH OF QUAY < TM_QUAYLENGTH

  @  Crane-allocation (when a ship arrives) @
  FOR EACH SHIP IN QUAY WITH SHIP_CRANE_ALLOC = FALSE
    THIS BERTH         ← RIGHTBERTH
    THIS SHIPCLASS     ← RIGHTCLASS
    THIS TERMINAL      ← RIGHTTERM
    SHIP_UNLOAD_RATE   ← 0
    SHIP_LOAD_RATE     ← 0
    SHIP_CRANE_SUPPLY ← 0
    FOR EACH CRANE IN BERTH_CRANES WITH CRANE_AVAILABILITY = TRUE
      SHIP_CRANE_SUPPLY ← SHIP_CRANE_SUPPLY + 1
    END
    THIS CRANE ← FIRST OF BERTH_CRANES
    J ← 0
    WHILE (J < MIN(CLASS_CRANE_DEMAND,SHIP_CRANE_SUPPLY)) ^
      (J ≤ LENGTH OF BERTH_CRANES)
      IF CRANE_AVAILABILITY = TRUE
        SHIP_UNLOAD_RATE    ← SHIP_UNLOAD_RATE + CRANE_UNLOADCAP x
                              SHIP_FORTYFEETFAC_IMP
        SHIP_LOAD_RATE      ← SHIP_LOAD_RATE + CRANE_LOADCAP x
                              SHIP_FORTYFEETFAC_EXP
        CRANE_MYSHIP        ← THIS SHIP
        CRANE_AVAILABILITY ← FALSE
        ACTIVATE THIS CRANE FROM CRANESTART IN CRANEMOD
        J ← J + 1
      END
      THIS CRANE ← SUCC OF THIS CRANE IN BERTH_CRANES
      IF (J ≤ LENGTH OF BERTH_CRANES)
    END
    SHIP_LOAD_RATE   ← SHIP_LOAD_RATE x 40÷23.5 IF SHIP_CARGOTYPE = 203
    SHIP_UNLOAD_RATE ← SHIP_UNLOAD_RATE x 40.1÷23.5 IF SHIP_CARGOTYPE=203
    SHIP_CRANE_ALLOC ← TRUE
    REACTIVATE THIS SHIP
  END
  REPEAT FROM TM_START

TM_REALLOCATE:
  @  Crane-reallocation (when a ship departs) @
  @  not applied for pontianak  @
  REPEAT FROM TM_START

SHIP_LOAD_RATE   ← SHIP_LOAD_RATE x 15÷23.5 IF (SHIP_CARGOTYPE = 201)^
(BERTH_NUMBER=4)
SHIP_UNLOAD_RATE ← SHIP_UNLOAD_RATE x 15÷23.5 IF (SHIP_CARGOTYPE=201)^
(BERTH_NUMBER=4)
```

```
MODEL HP2_11P                                                Date:  93/08/16
MOD CRANEMOD                                                 Time:  18:29:45

@@@@@@@@@@@@@@@@@@@@@@@@
@  PROCESS OF A CRANE  @
@@@@@@@@@@@@@@@@@@@@@@@@

CRANESTART:
   CRANE_AVAILABILITY ← FALSE
   CRANE_RESTTIME     ← CRANE_RESTTIME + NOW - CRANECALL
   CRANE_PREV_SHIP    ← NONE

CRANEWORK:
   CRANECALL ← NOW
   WAIT WHILE (TERM_SHIFT OF RIGHTTERM OF CRANE_MYSHIP = FALSE) v
   (STRIKE_ALARM = TRUE) v (TYPHOON_ALARM = TRUE)
   CRANE_DELAYTIME ← CRANE_DELAYTIME + NOW - CRANECALL
   CRANECALL ← NOW
   WAIT WHILE (TERM_SHIFT OF RIGHTTERM OF CRANE_MYSHIP = TRUE) ^
   (STRIKE_ALARM = FALSE) ^ ((CRANE_WORKTIME_A + NOW - CRANECALL) <
   CRANE_BREAKDOWN_TIME) ^ (TYPHOON_ALARM = FALSE) ^ ((SHIP_LOADING
   OF CRANE_MYSHIP = TRUE) v (SHIP_UNLOADING OF CRANE_MYSHIP = TRUE)) ^
   (SHIP_SHIFTED OF CRANE_MYSHIP = FALSE)
   CRANE_WORKTIME_A ← CRANE_WORKTIME_A + NOW - CRANECALL
   CRANE_WORKTIME_B ← CRANE_WORKTIME_B + NOW - CRANECALL
   GOTO CRANEBREAKDOWN IF CRANE_WORKTIME_A ≥ CRANE_BREAKDOWN_TIME
   GOTO CRANEREST IF ((SHIP_LOADING OF CRANE_MYSHIP = FALSE) ^
   (SHIP_UNLOADING OF CRANE_MYSHIP = FALSE)) v (SHIP_SHIFTED OF
   CRANE_MYSHIP = TRUE)
   REPEAT FROM CRANEWORK

CRANEBREAKDOWN:
   THIS SHIP     ← CRANE_MYSHIP
   THIS BERTH    ← RIGHTBERTH
   THIS TERMINAL ← RIGHTTERM
   IF SHIP_UNLOADING = TRUE
      SHIP_RATE      ← SHIP_RATE - CRANE_UNLOADCAP
      SHIP_LOAD_RATE ← SHIP_LOAD_RATE - CRANE_LOADCAP
   END
   IF SHIP_LOADING = TRUE
      SHIP_RATE ← SHIP_RATE - CRANE_LOADCAP
   END
   WAIT CRANE_BREAKDOWN_DURATION HOURS
   THIS SHIP       ← CRANE_MYSHIP IF CRANE_PREV_SHIP IS NONE
```

```
35     THIS SHIP      ← CRANE_PREV_SHIP IF CRANE_PREV_SHIP IS NOT NONE
36     THIS BERTH     ← RIGHTBERTH
37     THIS SHIPCLASS ← RIGHTCLASS
38     THIS TERMINAL  ← RIGHTTERM
39     CRANE_DOWNTIME ← CRANE_DOWNTIME + CRANE_BREAKDOWN_DURATION ÷ 24
       IF SHIPCALL2 = 0
40     CRANE_DOWNTIME ← CRANE_DOWNTIME + (CRANE_BREAKDOWN_DURATION ÷ 24) -
       NOW - SHIPCALL2 IF SHIPCALL2 > 0
41     CLASSDELAY_BREAKDOWN ← CLASSDELAY_BREAKDOWN +
       CRANE_BREAKDOWN_DURATION ÷ 24 IF SHIPCALL2 = 0
42     CLASSDELAY_BREAKDOWN ← CLASSDELAY_BREAKDOWN +
       (CRANE_BREAKDOWN_DURATION ÷ 24) - NOW - SHIPCALL2 IF SHIPCALL2 > 0
43     BERTHDELAY_BREAKDOWN ← BERTHDELAY_BREAKDOWN +
       CRANE_BREAKDOWN_DURATION ÷ 24
       IF (SHIPCALL2 = 0) ^ (BERTH_SINGLE = TRUE)
44     BERTHDELAY_BREAKDOWN ← BERTHDELAY_BREAKDOWN +
       (CRANE_BREAKDOWN_DURATION ÷ 24) - NOW - SHIPCALL2
       IF (SHIPCALL2 > 0) ^ (BERTH_SINGLE = TRUE)
45     BERTHDELAY_BREAKDOWN ← BERTHDELAY_BREAKDOWN + SHIP_LENGTH_BERTHED x
       CRANE_BREAKDOWN_DURATION ÷ 24
       IF (SHIPCALL2 = 0) ^ (BERTH_SINGLE = FALSE)
46     BERTHDELAY_BREAKDOWN ← BERTHDELAY_BREAKDOWN + SHIP_LENGTH_BERTHED x
       (CRANE_BREAKDOWN_DURATION ÷ 24) - NOW - SHIPCALL2
       IF (SHIPCALL2 > 0) ^ (BERTH_SINGLE = FALSE)
47     CRANE_BREAKDOWN_DURATION ← NORM_BD
48     CRANE_BREAKDOWN_TIME     ← EXP_IAT_B
49     CRANE_WORKTIME_A         ← 0
50     TERM_BREAKDOWNS          ← TERM_BREAKDOWNS + 1
51     THIS SHIP      ← CRANE_MYSHIP
52     THIS BERTH     ← RIGHTBERTH
53     THIS SHIPCLASS ← RIGHTCLASS
54     THIS TERMINAL  ← RIGHTTERM
55 @    IF CRANE_PREV_SHIP IS NOT NONE
56 @       CRANE_X_MIN          ← SHIP_X_BOW
57 @       CRANE_X_MAX          ← SHIP_X_STERN
58 @       CRANE_RESTTIME       ← CRANE_RESTTIME + SHIPCALL3 - SHIPCALL2 OF
           CRANE_PREV_SHIP
59 @       CRANE_DOWNTIME       ← CRANE_DOWNTIME + NOW - SHIPCALL3
60 @       CLASSDELAY_BREAKDOWN ← CLASSDELAY_BREAKDOWN + NOW - SHIPCALL3
61 @       BERTHDELAY_BREAKDOWN ← BERTHDELAY_BREAKDOWN + NOW - SHIPCALL3
           IF BERTH_SINGLE = TRUE
62 @       BERTHDELAY_BREAKDOWN ← BERTHDELAY_BREAKDOWN + SHIP_LENGTH_BERTHED x
           NOW - SHIPCALL3 IF BERTH_SINGLE = FALSE
63 @    END
```

```
@   CRANE_PREV_SHIP ← NONE
   IF (SHIP_UNLOADING = TRUE) ^ (SHIP_SHIFTED = FALSE)
      SHIP_RATE      ← SHIP_RATE + CRANE_UNLOADCAP
      SHIP_LOAD_RATE ← SHIP_LOAD_RATE + CRANE_LOADCAP
      REPEAT FROM CRANEWORK
   END
   IF (SHIP_LOADING = TRUE) ^ (SHIP_SHIFTED = FALSE)
      SHIP_RATE ← SHIP_RATE + CRANE_LOADCAP
      REPEAT FROM CRANEWORK
   END

CRANEREST:
   CRANECALL          ← NOW
   CRANE_AVAILABILITY ← TRUE
   CRANE_MYSHIP       ← NONE
   PASSIVATE
```

```
MODEL HP2_11P                                                       Date:  93/08/17
MOD SHIPMOD                                                         Time:  12:30:03

@@@@@@@@@@@@@@@@@@@@@@@
@  PROCESS OF A SHIP  @
@@@@@@@@@@@@@@@@@@@@@@@

SHIPSTART:
   CALL ARRIVALPATTERN
   SHIP_DAY ← 0
   IF STRIKE_ALARM = TRUE
      STRIKE_SPELL ← FLOOR(STRIKE_CALL + STRIKE_DURATION) - FLOOR(NOW)
      CALL PATTERNCHANGE(STRIKE_SPELL)
   END
   IF TYPHOON_ALARM = TRUE
      TYPHOON_SPELL ← FLOOR(TYPHOON_CALL + TYPHOON_DURATION) - FLOOR(NOW)
      CALL PATTERNCHANGE(TYPHOON_SPELL)
   END
   ENTER ARRIVINGSHIPS
   WAIT LENGTH_ARRIVALPATT
   WAIT SHIP_EXTRA_ARR
   LEAVE ARRIVINGSHIPS

   ENTER ROW
   ENTER PRIORITY_ROW IF CLASS_PRIORITY OF RIGHTCLASS = TRUE
   ENTER TERM_SHIPS OF RIGHTTERM
   TERM_ALLSHIPS OF RIGHTTERM ← TERM_ALLSHIPS OF RIGHTTERM + 1
   PASSIVATE
SHIPDELAY0:
   SHIPCALL1 ← NOW
   WAIT WHILE TYPHOON_ALARM = TRUE
   THIS BERTH          ← RIGHTBERTH
   THIS SHIPCLASS      ← RIGHTCLASS
   CLASSWAIT_TYPHOON ← CLASSWAIT_TYPHOON + NOW - SHIPCALL1
   BERTHWAIT_TYPHOON ← BERTHWAIT_TYPHOON + NOW - SHIPCALL1
   IF BERTH_SINGLE = TRUE
   BERTHWAIT_TYPHOON ← BERTHWAIT_TYPHOON + SHIP_LENGTH_BERTHED x
   NOW - SHIPCALL1 IF BERTH_SINGLE = FALSE
   SHIPCALL1 ← NOW
   WAIT WHILE (SHIP_DRAUGHT_BERTHED > ENTRANCE_DEPTH) v
   (SHIP_DRAUGHT_BERTHED >  MEAN_DEPTH +
   TIDE_AMPLITUDE_1 x COS(((NOW + 0.5 x SHIP_MOORINGTIME) x
   (2 x 3.141593) ÷ TIDE_PERIOD_1) - TIDE_ALPHA_1) +
    (2 x 3.141593) ÷ TIDE_PERIOD_2) - TIDE_ALPHA_2)) v
    (SHIP_DRAUGHT_BERTHED >  MEAN_DEPTH +
    TIDE_AMPLITUDE_1 x COS(((NOW + 0.25 x SHIP_MOORINGTIME) x
    (2 x 3.141593) ÷ TIDE_PERIOD_1) - TIDE_ALPHA_1) +
    TIDE_AMPLITUDE_2 x COS(((NOW + 0.25 x SHIP_MOORINGTIME) x
    (2 x 3.141593) ÷ TIDE_PERIOD_2) - TIDE_ALPHA_2))
    THIS BERTH     ← RIGHTBERTH
    THIS SHIPCLASS ← RIGHTCLASS
    CLASSWAIT_TIDE ← CLASSWAIT_TIDE + NOW - SHIPCALL1
    BERTHWAIT_TIDE ← BERTHWAIT_TIDE + NOW - SHIPCALL1
    IF BERTH_SINGLE = TRUE
    BERTHWAIT_TIDE ← BERTHWAIT_TIDE + SHIP_LENGTH_BERTHED x
    NOW - SHIPCALL1 IF BERTH_SINGLE = FALSE
    REPEAT FROM SHIPDELAY0 IF TYPHOON_ALARM = TRUE
    LEAVE ROW
    LEAVE PRIORITY_ROW IF CLASS_PRIORITY OF RIGHTCLASS = TRUE

    ENTER PORT
    WAIT MAX(0,SHIP_MOORINGTIME - 0.2) HOURS
    WAIT WHILE (MOORING[BERTH_NUMBER OF RIGHTBERTH + 2] = TRUE) v
    (MOORING[BERTH_NUMBER OF RIGHTBERTH + 1] = TRUE) v
    (MOORING[BERTH_NUMBER OF RIGHTBERTH] = TRUE)
    SHIP_MOORING_EXTRA ← NOW - QUEUETIME IN PORT +
    MAX(0,(SHIP_MOORINGTIME - 0.2)÷24)
    MOORING[BERTH_NUMBER OF RIGHTBERTH + 1] ← TRUE
    WAIT 0.2 HOURS
    MOORING[BERTH_NUMBER OF RIGHTBERTH + 1] ← FALSE
SHIPDELAY1:
    SHIPCALL1 ← NOW
    WAIT WHILE STRIKE_ALARM = TRUE
    THIS BERTH         ← RIGHTBERTH
    THIS TERMINAL      ← RIGHTTERM
    THIS SHIPCLASS     ← RIGHTCLASS
    CLASSDELAY_STRIKE ← CLASSDELAY_STRIKE  + (NOW-SHIPCALL1) x TERM_NET_FAC
    BERTHDELAY_STRIKE ← BERTHDELAY_STRIKE  + (NOW-SHIPCALL1) x TERM_NET_FAC
    IF BERTH_SINGLE = TRUE
    BERTHDELAY_STRIKE ← BERTHDELAY_STRIKE  + (NOW-SHIPCALL1) x TERM_NET_FAC
    x SHIP_LENGTH_BERTHED IF BERTH_SINGLE = FALSE
    SHIPCALL1 ← NOW
    WAIT WHILE TYPHOON_ALARM = TRUE
    THIS BERTH         ← RIGHTBERTH
    THIS TERMINAL      ← RIGHTTERM
    THIS SHIPCLASS     ← RIGHTCLASS
    CLASSDELAY_TYPHOON ← CLASSDELAY_TYPHOON + (NOW-SHIPCALL1)xTERM_NET_FAC
    BERTHDELAY_TYPHOON ← BERTHDELAY_TYPHOON + (NOW-SHIPCALL1)xTERM_NET_FAC
    IF BERTH_SINGLE = TRUE
    BERTHDELAY_TYPHOON ← BERTHDELAY_TYPHOON + (NOW-SHIPCALL1)xTERM_NET_FAC
    x SHIP_LENGTH_BERTHED IF BERTH_SINGLE = FALSE
    REPEAT FROM SHIPDELAY1 IF STRIKE_ALARM = TRUE

    ENTER QUAY
    SHIPCALL3 ← NOW
    SHIP_UNLOADING ← TRUE
    PASSIVATE
    SHIP_RATE ← SHIP_UNLOAD_RATE + CLASS_GEAR_UNLOADCAP OF RIGHTCLASS x 24
UNLOADING:
    SHIPCALL1 ← NOW   @  Required in case simulation-run ends  @
    WAIT WHILE (STRIKE_ALARM = TRUE) v (TERM_SHIFT OF RIGHTTERM = FALSE) v
    (TYPHOON_ALARM = TRUE)
    SHIPCALL1 ← NOW
    INTEGRATE WHILE (SHIP_LOAD < SHIP_IMPORTTOTAL) ^
    (STRIKE_ALARM = FALSE) ^ (TYPHOON_ALARM = FALSE) ^ (TERM_SHIFT OF
    RIGHTTERM = TRUE) WITH ACCURACY 4
    THIS BERTH      ← RIGHTBERTH
    THIS SHIPCLASS  ← RIGHTCLASS
    CLASS_WORK_TIME ← CLASS_WORK_TIME + NOW-SHIPCALL1
    BERTH_WORK_TIME ← BERTH_WORK_TIME + NOW-SHIPCALL1 IF BERTH_SINGLE=TRUE
    BERTH_WORK_TIME ← BERTH_WORK_TIME + SHIP_LENGTH_BERTHED x NOW-SHIPCALL1
    IF BERTH_SINGLE = FALSE
    REPEAT FROM UNLOADING IF (STRIKE_ALARM = TRUE) v
    (TERM_SHIFT OF RIGHTTERM = FALSE) v (TYPHOON_ALARM = TRUE)
    THIS TERMINAL  ← RIGHTTERM
    THIS CARGOTYPE ← FIRST CARGOTYPE IN TERM_CARGO WITH CT_NUMBER =
    SHIP_CARGOTYPE
    CT_THROUGHPUT_IMP   ← CT_THROUGHPUT_IMP + SHIP_IMPORTTOTAL IF NOW > 100
    CT_STORAGE_IMP_COV  ← CT_STORAGE_IMP_COV + SHIP_IMPORTTOTAL x
    (1 - CT_PERC_DEP_DIRECT ÷ 100) x SHIP_COVER_FAC ÷ 100
    CT_STORAGE_IMP_OPEN ← CT_STORAGE_IMP_OPEN + SHIP_IMPORTTOTAL x
    (1 - CT_PERC_DEP_DIRECT ÷ 100) x (1 - CLASS_FEEDER_PERC ÷ 100) x
    (1 - SHIP_COVER_FAC ÷ 100)
    CT_STORAGE_FEEDER   ← CT_STORAGE_FEEDER + SHIP_IMPORTTOTAL x
    (1 - CT_PERC_DEP_DIRECT ÷ 100) x CLASS_FEEDER_PERC ÷ 100

    SHIP_LOADING   ← TRUE
    SHIP_UNLOADING ← FALSE
    SHIP_RATE      ← SHIP_LOAD_RATE + CLASS_GEAR_LOADCAP OF RIGHTCLASS x 24
LOADING:
   WAIT WHILE (STRIKE_ALARM = TRUE) v (TERM_SHIFT OF RIGHTTERM = FALSE) v
   (TYPHOON_ALARM = TRUE)
   SHIPCALL1 ← NOW
   INTEGRATE WHILE (SHIP_LOAD < SHIP_IMPORTTOTAL + SHIP_EXPORTTOTAL) ^
   (STRIKE_ALARM = FALSE) ^ (TYPHOON_ALARM = FALSE) ^ (TERM_SHIFT OF
   RIGHTTERM = TRUE) WITH ACCURACY 4
   THIS BERTH       ← RIGHTBERTH
   THIS SHIPCLASS  ← RIGHTCLASS
   CLASS_WORK_TIME ← CLASS_WORK_TIME + NOW-SHIPCALL1
   BERTH_WORK_TIME ← BERTH_WORK_TIME + NOW-SHIPCALL1 IF BERTH_SINGLE=TRUE
   BERTH_WORK_TIME ← BERTH_WORK_TIME + SHIP_LENGTH_BERTHED x NOW-SHIPCALL1
   IF BERTH_SINGLE = FALSE
   REPEAT FROM LOADING IF (STRIKE_ALARM = TRUE) v
   (TERM_SHIFT OF RIGHTTERM = FALSE) v (TYPHOON_ALARM = TRUE)
   THIS TERMINAL  ← RIGHTTERM
   THIS CARGOTYPE ← FIRST CARGOTYPE IN TERM_CARGO WITH CT_NUMBER =
   SHIP_CARGOTYPE
   CT_THROUGHPUT_EXP   ← CT_THROUGHPUT_EXP + SHIP_EXPORTTOTAL IF NOW > 100
   CT_STORAGE_EXP_COV  ← CT_STORAGE_EXP_COV  - SHIP_EXPORTTOTAL x
   (1 - CT_PERC_ARR_DIRECT ÷ 100) x SHIP_COVER_FAC ÷ 100
   CT_STORAGE_EXP_OPEN ← CT_STORAGE_EXP_OPEN - SHIP_EXPORTTOTAL x
   (1 - CT_PERC_ARR_DIRECT ÷ 100) x (1 - CLASS_FEEDER_PERC ÷ 100) x
   (1 - SHIP_COVER_FAC ÷ 100)
   CT_STORAGE_FEEDER   ← MAX(0, CT_STORAGE_FEEDER - SHIP_EXPORTTOTAL x
   (1 - CT_PERC_ARR_DIRECT ÷ 100) x CLASS_FEEDER_PERC ÷ 100)
   IF SHIP_CARGOTYPE = 205   @  for pontianak
      THIS CARGOTYPE ← FIRST CARGOTYPE IN TERM_CARGO WITH CT_NUMBER = 101
      CT_STORAGE_EXP_OPEN ← CT_STORAGE_EXP_OPEN - (SHIP_EXPORTTOTAL +
      SHIP_IMPORTTOTAL) ÷ 2 x 9.6
   END
   SHIP_LOADING ← FALSE

   ENTER DEPARTINGSHIPS
   CALL DEPARTUREPATTERN
   SHIP_DAY ← 0
   LEAVE BERTH_SHIPS OF RIGHTBERTH
   LEAVE QUAY
   SHIPCALL2 ← NOW
SHIPDELAY2:
   SHIPCALL1 ← NOW
   WAIT WHILE TYPHOON_ALARM = TRUE
   THIS BERTH          ← RIGHTBERTH
   THIS SHIPCLASS      ← RIGHTCLASS
   CLASSDELAY_TYPHOON ← CLASSDELAY_TYPHOON + NOW - SHIPCALL1
```

```
132    BERTHDELAY_TYPHOON ← BERTHDELAY_TYPHOON + NOW - SHIPCALL1
       IF BERTH_SINGLE = TRUE
133    BERTHDELAY_TYPHOON ← BERTHDELAY_TYPHOON + SHIP_LENGTH_BERTHED x
       NOW - SHIPCALL1 IF BERTH_SINGLE = FALSE
134    BERTHDELAY_TYPHOON ← BERTHDELAY_TYPHOON + NOW - SHIPCALL1
       IF BERTH_SINGLE = TRUE
135    BERTHDELAY_TYPHOON ← BERTHDELAY_TYPHOON + SHIP_LENGTH_BERTHED x
       NOW - SHIPCALL1 IF BERTH_SINGLE = FALSE
136    SHIPCALL1 ← NOW
137    WAIT WHILE (MOORING[BERTH_NUMBER OF RIGHTBERTH + 2] = TRUE) v
       (MOORING[BERTH_NUMBER OF RIGHTBERTH + 1] = TRUE) v
       (MOORING[BERTH_NUMBER OF RIGHTBERTH] = TRUE)
138    CLASSDELAY_LEAVING ← CLASSDELAY_LEAVING + NOW - SHIPCALL1
139    BERTHDELAY_LEAVING ← BERTHDELAY_LEAVING + NOW - SHIPCALL1
       IF BERTH_SINGLE = TRUE
140    BERTHDELAY_LEAVING ← BERTHDELAY_LEAVING + SHIP_LENGTH_BERTHED x
       NOW - SHIPCALL1 IF BERTH_SINGLE = FALSE
141    REPEAT FROM SHIPDELAY2 IF TYPHOON_ALARM = TRUE
142
143    THIS BERTH         ← RIGHTBERTH
144    THIS SHIPCLASS     ← RIGHTCLASS
145    BERTH_LENGTH_FREE ← BERTH_LENGTH_FREE + SHIP_LENGTH_BERTHED
146    CLASS_ALLSHIPS     ← CLASS_ALLSHIPS + 1
147    BERTH_OCC_TIME     ← BERTH_OCC_TIME + NOW - QUEUETIME IN PORT +
       (SHIP_MOORINGTIME ÷ 24) + SHIP_MOORING_EXTRA IF BERTH_SINGLE = TRUE
148    BERTH_OCC_TIME     ← BERTH_OCC_TIME + SHIP_LENGTH_BERTHED x NOW -
       QUEUETIME IN PORT + (SHIP_MOORINGTIME ÷ 24) + SHIP_MOORING_EXTRA
       IF BERTH_SINGLE = FALSE
149    CLASS_BERTHTIME    ← CLASS_BERTHTIME + NOW - QUEUETIME IN PORT +
       (SHIP_MOORINGTIME ÷ 24) + SHIP_MOORING_EXTRA
150    CLASS_WAITINGTIME ← CLASS_WAITINGTIME + QUEUETIME IN PORT - QUEUETIME
       IN TERM_SHIPS OF RIGHTTERM
151    CLASS_MOORINGTIME ← CLASS_MOORINGTIME + (SHIP_MOORINGTIME ÷ 24) +
       SHIP_MOORING_EXTRA
152    LEAVE TERM_SHIPS OF RIGHTTERM
153    MOORING[BERTH_NUMBER OF RIGHTBERTH + 1] ← TRUE
154    WAIT 0.2 HOURS
155    MOORING[BERTH_NUMBER OF RIGHTBERTH + 1] ← FALSE
156    LEAVE PORT
157    WAIT MAX(0,SHIP_MOORINGTIME - 0.2) HOURS
158    ENTER BUOY
159    WAIT WHILE (SHIP_DRAUGHT_BERTHED > MEAN_DEPTH + TIDE_AMPLITUDE_1 x
       COS(((NOW + 0.5 x SHIP_MOORINGTIME) x (2 x 3.141593) ÷ TIDE_PERIOD_1) -
x (2 x 3.141593) ÷ TIDE_PERIOD_2) - TIDE_ALPHA_2)) v
(SHIP_DRAUGHT_BERTHED >  MEAN_DEPTH + TIDE_AMPLITUDE_1 x
COS(((NOW + 0.75 x SHIP_MOORINGTIME) x (2 x 3.141593) ÷ TIDE_PERIOD_1)-
TIDE_ALPHA_1) + TIDE_AMPLITUDE_2 x COS(((NOW + 0.75 x SHIP_MOORINGTIME)
x (2 x 3.141593) ÷ TIDE_PERIOD_2) - TIDE_ALPHA_2))
THIS BERTH          ← RIGHTBERTH
THIS SHIPCLASS      ← RIGHTCLASS
CLASSDELAY_TIDE     ← CLASSDELAY_TIDE + NOW - QUEUETIME IN BUOY
LEAVE BUOY
WAIT LENGTH_DEPARTUREPATT - NOW - SHIPCALL2
WAIT SHIP_EXTRA_DEP
LEAVE DEPARTINGSHIPS
TERMINATE
```

```
MODEL HP2_11P                                                         Date:  93/08/16
MOD SHIFTMOD                                                          Time:  18:31:35
```

```
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@  PROCESS OF THE SHIFTS ON EACH TERMINAL  @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

SHIFTSTART:
   WAIT SHIFT_TIME[1] HOURS

SHIFTREPEAT:
   WAIT 1 IF (((DAYNUMBER = 5) ^ (TERM_SAT_LOAD OF SHIFT_TERM = FALSE)) ^
   (TERM_SUN_LOAD OF SHIFT_TERM = TRUE)) v
   (((DAYNUMBER = 6) ^ (TERM_SUN_LOAD OF SHIFT_TERM = FALSE)) ^
   (TERM_SAT_LOAD OF SHIFT_TERM = TRUE))
   WAIT 2 IF (DAYNUMBER = 5) ^ (TERM_SAT_LOAD OF SHIFT_TERM = FALSE) ^
   (TERM_SUN_LOAD OF SHIFT_TERM = FALSE)
   FOR K ← 1 TO TERM_SHIFTS OF SHIFT_TERM
      TERM_SHIFT OF SHIFT_TERM ← TRUE
      WAIT SHIFT_DURATION[K] HOURS
      TERM_SHIFT OF SHIFT_TERM ← FALSE
      WAIT (SHIFT_TIME[K + 1] - SHIFT_TIME[K] + SHIFT_DURATION[K]) HOURS
      IF K < TERM_SHIFTS OF SHIFT_TERM
      WAIT ((SHIFT_TIME[1]+24) - SHIFT_TIME[K] + SHIFT_DURATION[K]) HOURS
      IF K = TERM_SHIFTS OF SHIFT_TERM
   END
   REPEAT FROM SHIFTREPEAT
```

```
MODEL HP2_11P                                               Date:  93/08/16
MOD STORAGEMOD                                              Time:  18:31:37

@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@  PROCESS OF THE STORAGEMASTER  @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

SM_START:
  @  Handling of cargo arriving and departing with inland transport  @
  WAIT 1 DAY
  FOR EACH TERMINAL IN TERMINALSET
    FOR EACH CARGOTYPE IN TERM_CARGO
      CT_EXP_ARR_OPEN ← 0
      CT_EXP_ARR_COV  ← 0
      CT_IMP_DEP_OPEN ← 0
      CT_IMP_DEP_COV  ← 0
    END
  END
  DAYNUMBER ← DAYNUMBER + 1
  DAYNUMBER ← 0 IF DAYNUMBER = 7
  REPEAT FROM SM_START IF (STRIKE_ALARM = TRUE) v (TYPHOON_ALARM = TRUE)
  @  Arrival of export cargo  @
  FOR EACH SHIP IN ARRIVINGSHIPS
    THIS SHIPCLASS ← RIGHTCLASS
    THIS CARGOTYPE ← FIRST CARGOTYPE IN TERM_CARGO OF RIGHTTERM WITH
    CT_NUMBER = SHIP_CARGOTYPE
    SHIP_DAY ← SHIP_DAY + 1
    CT_EXP_ARR_COV ← CT_EXP_ARR_COV + SHIP_EXPORTTOTAL x
    (SHIP_COVER_FAC ÷ 100) x SHIP_PERC_ARR[SHIP_DAY] ÷ 100
    CT_EXP_ARR_OPEN ← CT_EXP_ARR_OPEN + SHIP_EXPORTTOTAL x
    (1 - SHIP_COVER_FAC ÷ 100) x (1 - CLASS_FEEDER_PERC ÷ 100) x
    SHIP_PERC_ARR[SHIP_DAY] ÷ 100
    IF (SHIP_DAY = 15) ^ (SHIP_CARGOTYPE = 205)    @  for pontianak
       THIS CARGOTYPE ← FIRST CARGOTYPE IN TERM_CARGO OF RIGHTTERM WITH
       CT_NUMBER = 101
       CT_EXP_ARR_OPEN ← CT_EXP_ARR_OPEN + (SHIP_EXPORTTOTAL +
       SHIP_IMPORTTOTAL) ÷ 2 x 9.6
    END
  END
  @  Departure of import cargo  @
  FOR EACH SHIP IN DEPARTINGSHIPS
    THIS TERMINAL  ← RIGHTTERM
    THIS SHIPCLASS ← RIGHTCLASS
    THIS CARGOTYPE ← FIRST CARGOTYPE IN TERM_CARGO WITH CT_NUMBER =
  SHIP_CARGOTYPE
  GOTO NEXT IF ((DAYNUMBER = 6) ^ (TERM_SAT_TRANS = FALSE)) v
  ((DAYNUMBER = 0) ^ (TERM_SUN_TRANS = FALSE))
  SHIP_DAY ← SHIP_DAY + 1
  CT_IMP_DEP_COV ← CT_IMP_DEP_COV + SHIP_IMPORTTOTAL x
  (SHIP_COVER_FAC ÷ 100) x SHIP_PERC_DEP[SHIP_DAY] ÷ 100
  CT_IMP_DEP_OPEN ← CT_IMP_DEP_OPEN + SHIP_IMPORTTOTAL x
  (1 - SHIP_COVER_FAC ÷ 100) x (1 - CLASS_FEEDER_PERC ÷ 100) x
  SHIP_PERC_DEP[SHIP_DAY] ÷ 100
  NEXT:
END
@  Notebook  @
FOR EACH TERMINAL IN TERMINALSET
  TERM_AREA_OCCUPIED ← 0
  TERM_VOL_OCCUPIED  ← 0
  FOR EACH CARGOTYPE IN TERM_CARGO
    CT_STORAGE_EXP_COV  ← CT_STORAGE_EXP_COV  + CT_EXP_ARR_COV
    CT_STORAGE_EXP_OPEN ← CT_STORAGE_EXP_OPEN + CT_EXP_ARR_OPEN
    CT_STORAGE_IMP_COV  ← CT_STORAGE_IMP_COV  - CT_IMP_DEP_COV
    CT_STORAGE_IMP_OPEN ← CT_STORAGE_IMP_OPEN - CT_IMP_DEP_OPEN
    IF NOW > 100
      IF CT_CODE = "A"
        CT_STACK_EXP     ← (CT_STORAGE_EXP_OPEN + CT_STORAGE_FEEDER) x
        1 - CT_EMPTIESPERC_EXP ÷ 100
        CT_STACK_IMP     ← CT_STORAGE_IMP_OPEN x
        1 - CT_EMPTIESPERC_IMP ÷ 100
        CT_STACK_EMP_EXP ← (CT_STORAGE_EXP_OPEN + CT_STORAGE_FEEDER) x
        CT_EMPTIESPERC_EXP ÷ 100
        CT_STACK_EMP_IMP ← CT_STORAGE_IMP_OPEN x
        CT_EMPTIESPERC_IMP ÷ 100
        CT_GRSLOT_EXP     ← CT_STACK_EXP ÷ CT_STACKHEIGHT_EXP
        CT_GRSLOT_IMP     ← CT_STACK_IMP ÷ CT_STACKHEIGHT_IMP
        CT_GRSLOT_EMP     ← (CT_STACK_EMP_EXP ÷ CT_STACKHEIGHT_EMP) +
        CT_STACK_EMP_IMP ÷ CT_STACKHEIGHT_EMP
        CT_STORAGE_AREA_OPEN ← (CT_GRSLOT_EXP + CT_GRSLOT_IMP +
        CT_GRSLOT_EMP) x (2.44 x 6.10) x CT_GROSS_FAC
        STORE CT_GRSLOT_EXP
        AS TERMINAL_NUMBER¦" EXSL   "¦CT_NUMBER
        STORE CT_GRSLOT_IMP
        AS TERMINAL_NUMBER¦" IMSL   "¦CT_NUMBER
        STORE CT_GRSLOT_EMP
        AS TERMINAL_NUMBER¦" EMSL   "¦CT_NUMBER
        STORE CT_STORAGE_AREA_OPEN
        AS TERMINAL_NUMBER¦" OPEN   "¦CT_NUMBER
```

```
65           STORE CT_STORAGE_FEEDER
             AS TERMINAL_NUMBER¦" FEED    "¦CT_NUMBER
66           TERM_AREA_OCCUPIED ← TERM_AREA_OCCUPIED + CT_STORAGE_AREA_OPEN
67         END
68         IF CT_CODE = "B"
69           CT_STORAGE_AREA_COV  ← (CT_GROSS_FAC x
             CT_STORAGE_EXP_COV + CT_STORAGE_IMP_COV) ÷
             CT_DENSITY x CT_STACKHEIGHT_GEN_COV
70           CT_STORAGE_AREA_OPEN ← (CT_GROSS_FAC x
             CT_STORAGE_EXP_OPEN + CT_STORAGE_IMP_OPEN) ÷
             CT_DENSITY x CT_STACKHEIGHT_GEN_OPEN
71           STORE CT_STORAGE_AREA_OPEN
             AS TERMINAL_NUMBER¦" OPEN    "¦CT_NUMBER
72           STORE CT_STORAGE_AREA_COV
             AS TERMINAL_NUMBER¦" COV    "¦CT_NUMBER
73           TERM_AREA_OCCUPIED ← TERM_AREA_OCCUPIED + CT_STORAGE_AREA_OPEN +
             CT_STORAGE_AREA_COV
74         END
75         IF (CT_CODE = "C") v (CT_CODE = "D")
76           CT_STORAGE_VOLUME ← ((CT_STORAGE_EXP_OPEN + CT_STORAGE_IMP_OPEN)
             x CT_GROSS_FAC) ÷ CT_DENSITY
77           STORE CT_STORAGE_VOLUME
             AS TERMINAL_NUMBER¦" VOL    "¦CT_NUMBER
78           TERM_VOL_OCCUPIED ← TERM_VOL_OCCUPIED + CT_STORAGE_VOLUME
79         END
80         STORE CT_EXP_ARR_OPEN + CT_EXP_ARR_COV
           AS TERMINAL_NUMBER¦" ARR    "¦CT_NUMBER
81         STORE CT_IMP_DEP_OPEN + CT_IMP_DEP_COV
           AS TERMINAL_NUMBER¦" DEP    "¦CT_NUMBER
82       END
83     END
84     STORE TERM_AREA_OCCUPIED ÷ TERM_AREA_CAPACITY
       AS TERMINAL_NUMBER¦" AREA_OR" IF (TERM_AREA_CAPACITY = 0) ^ (NOW>100)
85     STORE TERM_VOL_OCCUPIED ÷ TERM_VOL_CAPACITY
       AS TERMINAL_NUMBER¦" VOL_OR" IF (TERM_VOL_CAPACITY = 0) ^ (NOW>100)
86   END
87
88 REPEAT FROM SM_START
```

```
MODEL HP2_11P                                                       Date:  93/08/16
MOD STRIKEMOD                                                       Time:  18:31:47
```

```
@@@@@@@@@@@@@@@@@@@@@@@@@
@  PROCESS OF A STRIKE  @
@@@@@@@@@@@@@@@@@@@@@@@@@

STRIKESTART:
   @  Wait for strike  @
   STRIKE_ALARM ← FALSE
   STRIKE_INTERARRIVALTIME ← EXP_IAT_S
   WAIT STRIKE_INTERARRIVALTIME DAYS

   @  Strike start  @
   STRIKE_ALARM      ← TRUE
   STRIKE_DURATION   ← NORM_SD
   STRIKE_SPELL      ← FLOOR(NOW + STRIKE_DURATION) - FLOOR(NOW)
   FOR EACH SHIP IN ARRIVINGSHIPS
      CALL PATTERNCHANGE(STRIKE_SPELL)
   END
   FOR EACH SHIP IN DEPARTINGSHIPS
      SHIP_EXTRA_DEP ← SHIP_EXTRA_DEP + STRIKE_SPELL
      IF TERM_SAT_TRANS OF RIGHTTERM = FALSE
         SHIP_PERC_DEP[LENGTH_DEPARTUREPATT + 1] ← 0
         SHIP_EXTRA_DEP ← SHIP_EXTRA_DEP + 1
      END
      IF TERM_SUN_TRANS OF RIGHTTERM = FALSE
         SHIP_PERC_DEP[LENGTH_DEPARTUREPATT + 2] ← 0
         SHIP_EXTRA_DEP ← SHIP_EXTRA_DEP + 1
      END
   END
   STRIKE_CALL ← NOW
   WAIT STRIKE_DURATION DAYS

   @  Strike end  @
   NR_STRIKES ← NR_STRIKES + 1

   FOR EACH SHIP IN QUAY
      THIS BERTH        ← RIGHTBERTH
      THIS TERMINAL     ← RIGHTTERM
      THIS SHIPCLASS    ← RIGHTCLASS
      CLASSDELAY_STRIKE ← CLASSDELAY_STRIKE + STRIKE_DURATION x
      TERM_NET_FAC
      BERTHDELAY_STRIKE ← BERTHDELAY_STRIKE + STRIKE_DURATION x
      TERM_NET_FAC IF BERTH_SINGLE = TRUE
      BERTHDELAY_STRIKE ← BERTHDELAY_STRIKE + STRIKE_DURATION x
      TERM_NET_FAC x SHIP_LENGTH_BERTHED IF BERTH_SINGLE = FALSE
   END

   IF (TYPHOON_ALARM = TRUE) ^ (TYPHOON_CALL > STRIKE_CALL)
      FOR EACH SHIP IN QUAY
         THIS BERTH        ← RIGHTBERTH
         THIS TERMINAL     ← RIGHTTERM
         THIS SHIPCLASS    ← RIGHTCLASS
         CLASSDELAY_TYPHOON ← CLASSDELAY_TYPHOON - (NOW - TYPHOON_CALL) x
         TERM_NET_FAC
         BERTHDELAY_TYPHOON ← BERTHDELAY_TYPHOON - (NOW - TYPHOON_CALL) x
         TERM_NET_FAC IF BERTH_SINGLE = TRUE
         BERTHDELAY_TYPHOON ← BERTHDELAY_TYPHOON - (NOW - TYPHOON_CALL) x
         TERM_NET_FAC x SHIP_LENGTH_BERTHED IF BERTH_SINGLE = FALSE
      END
   END
   REPEAT FROM STRIKESTART
```

```
MODEL HP2_11P                                                  Date:  93/08/16
MOD TYPHOONMOD                                                 Time:  18:31:58

@@@@@@@@@@@@@@@@@@@@@@@@@@
@  PROCESS OF A TYPHOON  @
@@@@@@@@@@@@@@@@@@@@@@@@@@

TYPHOONSTART:
   @  Wait for typhoon  @
   TYPHOON_ALARM ← FALSE
   WAIT WHILE (TYPHOONBEGIN > 365 x ((NOW ÷ 365) - FLOOR(NOW ÷ 365))) v
   (TYPHOONEND < 365 x ((NOW ÷ 365) - FLOOR(NOW ÷ 365)))
   TYPHOON_INTERARRIVALTIME ← EXP_IAT_T
   WAIT TYPHOON_INTERARRIVALTIME DAYS

   @  Typhoon start  @
   TYPHOON_DURATION ← NORM_TD
   TYPHOON_ALARM    ← TRUE
   TYPHOON_SPELL    ← FLOOR(NOW + TYPHOON_DURATION) - FLOOR(NOW)
   FOR EACH SHIP IN ARRIVINGSHIPS
      CALL PATTERNCHANGE(TYPHOON_SPELL)
   END
   FOR EACH SHIP IN DEPARTINGSHIPS
      SHIP_EXTRA_DEP ← SHIP_EXTRA_DEP + TYPHOON_SPELL
      IF TERM_SAT_TRANS OF RIGHTTERM = FALSE
         SHIP_PERC_DEP[LENGTH_DEPARTUREPATT + 1] ← 0
         SHIP_EXTRA_DEP ← SHIP_EXTRA_DEP + 1
      END
      IF TERM_SUN_TRANS OF RIGHTTERM = FALSE
         SHIP_PERC_DEP[LENGTH_DEPARTUREPATT + 2] ← 0
         SHIP_EXTRA_DEP ← SHIP_EXTRA_DEP + 1
      END
   END
   TYPHOON_CALL ← NOW
   WAIT TYPHOON_DURATION DAYS

   @  Typhoon end  @
   NR_TYPHOONS ← NR_TYPHOONS + 1
   FOR EACH SHIP IN QUAY
      THIS BERTH          ← RIGHTBERTH
      THIS TERMINAL       ← RIGHTTERM
      THIS SHIPCLASS      ← RIGHTCLASS
      CLASSDELAY_TYPHOON ← CLASSDELAY_TYPHOON + TYPHOON_DURATION x
      TERM_NET_FAC
   BERTHDELAY_TYPHOON ← BERTHDELAY_TYPHOON + TYPHOON_DURATION x
   TERM_NET_FAC IF BERTH_SINGLE = TRUE
   BERTHDELAY_TYPHOON ← BERTHDELAY_TYPHOON + TYPHOON_DURATION x
   TERM_NET_FAC x SHIP_LENGTH_BERTHED IF BERTH_SINGLE = FALSE
END
IF (STRIKE_ALARM = TRUE) ^ (STRIKE_CALL > TYPHOON_CALL)
   FOR EACH SHIP IN QUAY
      THIS BERTH        ← RIGHTBERTH
      THIS TERMINAL     ← RIGHTTERM
      THIS SHIPCLASS    ← RIGHTCLASS
      CLASSDELAY_STRIKE ← CLASSDELAY_STRIKE - (NOW - STRIKE_CALL) x
      TERM_NET_FAC
      BERTHDELAY_STRIKE ← BERTHDELAY_STRIKE - (NOW - STRIKE_CALL) x
      TERM_NET_FAC IF BERTH_SINGLE = TRUE
      BERTHDELAY_STRIKE ← BERTHDELAY_STRIKE - (NOW - STRIKE_CALL) x
      TERM_NET_FAC x SHIP_LENGTH_BERTHED IF BERTH_SINGLE = FALSE
   END
END
REPEAT FROM TYPHOONSTART
```

```
MODEL HP2_11P                                                   Date:  93/08/16
MOD REPORTMOD                                                   Time:  18:33:35

@@@@@@@@@@@@@@@@@@@
@  REPORT MODULE  @
@@@@@@@@@@@@@@@@@@@

REPORTSTART:
  REWIND REPORT
  WRITE "########## PERFORMANCE OF THE TERMINALS ##########" TO REPORT
  WITH IMAGE A
  WRITE " " TO REPORT WITH IMAGE A
  I ← 0
  FOR EACH TERMINAL IN TERMINALSET
    I ← I + 1
    WRITE " " TO REPORT WITH IMAGE A
    WRITE "-----------" TO REPORT WITH IMAGE A
    WRITE "TERMINAL";TERMINAL_NUMBER TO REPORT WITH IMAGE A^xx
    WRITE "-----------" TO REPORT WITH IMAGE A
    WRITE " " TO REPORT WITH IMAGE A

    @  Inter-arrivaltime of ships at this terminal  @
    WRITE "AVERAGE ANNUAL NUMBER OF SHIPS AT THIS TERMINAL =";
    TERM_ALLSHIPS x 365 ÷ SIMULATIONTIME - LENGTH_ARRIVALPATT TO REPORT
    WITH IMAGE A^xxxx.xx
    WRITE " " TO REPORT WITH IMAGE A

    @  Throughput-data  @
    FOR EACH CARGOTYPE IN TERM_CARGO
      IF CT_CODE = "A"
        WRITE "ANNUAL THROUGHPUT OF";NAME OF THIS CARGOTYPE;"=";
        (CT_THROUGHPUT_EXP + CT_THROUGHPUT_IMP) x 365 ÷ SIMULATIONTIME -
        100; "TEU" TO REPORT WITH IMAGE A^A^A^xxxxxxx^A
        WRITE "  DISTRIBUTION OF OCCUPIED STORAGE-AREA FOR";NAME OF THIS
        CARGOTYPE;":" TO REPORT WITH IMAGE A^A^A
        WRITE "  SEE STORESTREAM";I;"OPEN";CT_NUMBER TO REPORT WITH
        IMAGE A^xxAxxx
        WRITE "  DISTRIBUTIONS OF OCCUPIED GROUNDSLOTS OF";NAME OF THIS
        CARGOTYPE;":" TO REPORT WITH IMAGE A^A^A
        WRITE "  EXPORT:  SEE STORESTREAM";I;"EXSL";CT_NUMBER TO REPORT
        WITH IMAGE A^xxAxxx
        WRITE "  IMPORT:  SEE STORESTREAM";I;"IMSL";CT_NUMBER TO REPORT
        WITH IMAGE A^xxAxxx
        WRITE "  EMPTIES:  SEE STORESTREAM";I;"EMSL";CT_NUMBER TO REPORT
```

```
            WITH IMAGE A^xxAxxx
32        END
33        IF CT_CODE = "B"
34          WRITE "ANNUAL THROUGHPUT OF";NAME OF THIS CARGOTYPE;"=";
            (CT_THROUGHPUT_EXP + CT_THROUGHPUT_IMP) x 365 ÷ SIMULATIONTIME -
            100;"TONS" TO REPORT WITH IMAGE A^A^A^xxxxxxx^A
35          WRITE "  DISTRIBUTION OF OCCUPIED OPEN STORAGE-AREA FOR";NAME OF
            THIS CARGOTYPE;":" TO REPORT WITH IMAGE A^A^A
36          WRITE "   SEE STORESTREAM";I;"OPEN";CT_NUMBER TO REPORT WITH
            IMAGE A^xxAxxx
37          WRITE "  DISTRIBUTION OF OCCUPIED COVERED STORAGE-AREA FOR";NAME
            OF THIS CARGOTYPE;":" TO REPORT WITH IMAGE A^A^A
38          WRITE "   SEE STORESTREAM";I;"COV";CT_NUMBER TO REPORT WITH
            IMAGE A^xxAxxx
39        END
40        IF (CT_CODE = "C") v (CT_CODE = "D")
41          WRITE "ANNUAL THROUGHPUT OF";NAME OF THIS CARGOTYPE;"=";
            (CT_THROUGHPUT_EXP + CT_THROUGHPUT_IMP) x 365 ÷ SIMULATIONTIME -
            100;"TONS" TO REPORT WITH IMAGE A^A^A^xxxxxxx^A
42          WRITE "  DISTRIBUTION OF STORAGE-VOLUME FOR";NAME OF
            THIS CARGOTYPE;":" TO REPORT WITH IMAGE A^A^A
43          WRITE "   SEE STORESTREAM";I;"VOL";CT_NUMBER TO REPORT WITH
            IMAGE A^xxAxxx
44        END
45        TERM_ARR_TRUCK ← TERM_ARR_TRUCK + ((CT_THROUGHPUT_EXP x
          CT_PERC_ARR_ROAD ÷ 100) ÷ CT_CAP_TRUCK) x 365 ÷ SIMULATIONTIME-100
46        TERM_DEP_TRUCK ← TERM_DEP_TRUCK + ((CT_THROUGHPUT_IMP x
          CT_PERC_DEP_ROAD ÷ 100) ÷ CT_CAP_TRUCK) x 365 ÷ SIMULATIONTIME-100
47        TERM_ARR_WAGON ← TERM_ARR_WAGON + ((CT_THROUGHPUT_EXP x
          CT_PERC_ARR_RAIL ÷ 100) ÷ CT_CAP_WAGON) x 365 ÷ SIMULATIONTIME-100
48        TERM_DEP_WAGON ← TERM_DEP_WAGON + ((CT_THROUGHPUT_IMP x
          CT_PERC_DEP_RAIL ÷ 100) ÷ CT_CAP_WAGON) x 365 ÷ SIMULATIONTIME-100
49        TERM_ARR_BARGE ← TERM_ARR_BARGE + ((CT_THROUGHPUT_EXP x
          CT_PERC_ARR_IWT ÷ 100) ÷ CT_CAP_BARGE) x 365 ÷ SIMULATIONTIME-100
50        TERM_DEP_BARGE ← TERM_DEP_BARGE + ((CT_THROUGHPUT_IMP x
          CT_PERC_DEP_IWT ÷ 100) ÷ CT_CAP_BARGE) x 365 ÷ SIMULATIONTIME-100
51        WRITE " " TO REPORT WITH IMAGE A
52      END
53      WRITE "DISTRIBUTIONS OF THE OCCUPANCY-RATE OF THE TOTAL STORAGE-AREA"
        TO REPORT WITH IMAGE A
54      WRITE "C.Q. THE TOTAL STORAGE-VOLUME" TO REPORT WITH IMAGE A
55      WRITE "SEE STORESTREAMS";I;"AREA_OR  AND ";I;"VOL_OR" TO REPORT WITH
        IMAGE A^xxA^xxA
56      WRITE " " TO REPORT WITH IMAGE A
```

```
@  Data of trucks\wagons\barges  @
WRITE "ANNUAL NUMBER OF TRUCKS LEAVING THIS TERMINAL     =";
TERM_DEP_TRUCK TO REPORT WITH IMAGE A^xxxxxxx
WRITE "ANNUAL NUMBER OF TRUCKS ENTERING THIS TERMINAL    =";
TERM_ARR_TRUCK TO REPORT WITH IMAGE A^xxxxxxx
WRITE "ANNUAL NUMBER OF WAGONS LEAVING THIS TERMINAL     =";
TERM_DEP_WAGON TO REPORT WITH IMAGE A^xxxxxxx
WRITE "ANNUAL NUMBER OF WAGONS ENTERING THIS TERMINAL    =";
TERM_ARR_WAGON TO REPORT WITH IMAGE A^xxxxxxx
WRITE "ANNUAL NUMBER OF BARGES LEAVING THIS TERMINAL     =";
TERM_DEP_BARGE TO REPORT WITH IMAGE A^xxxxxxx
WRITE "ANNUAL NUMBER OF BARGES ENTERING THIS TERMINAL    =";
TERM_ARR_BARGE TO REPORT WITH IMAGE A^xxxxxxx

SIMULATIONTIME ← SIMULATIONTIME - LENGTH_ARRIVALPATT
@  Performance of berths at this terminal  @
FOR EACH BERTH IN TERM_BERTHSET
  BERTHDELAY_SHIFTING ← 0 IF ABS(BERTHDELAY_SHIFTING) < 0.005
  BERTHDELAY_TYPHOON  ← 0 IF ABS(BERTHDELAY_TYPHOON)  < 0.005
  BERTHDELAY_LEAVING  ← 0 IF ABS(BERTHDELAY_LEAVING)  < 0.005
  BERTHDELAY_STRIKE   ← 0 IF ABS(BERTHDELAY_STRIKE)   < 0.005
  BERTHWAIT_TIDE      ← 0 IF ABS(BERTHWAIT_TIDE)      < 0.005
  BERTHWAIT_TYPHOON   ← 0 IF ABS(BERTHWAIT_TYPHOON)   < 0.005
  BERTH_OCC_TIME      ← BERTH_OCC_TIME     x 24 x 365 ÷ SIMULATIONTIME
  BERTH_WORK_TIME     ← BERTH_WORK_TIME    x 24 x 365 ÷ SIMULATIONTIME
  BERTHWAIT_TIDE      ← BERTHWAIT_TIDE     x 24 x 365 ÷ SIMULATIONTIME
  BERTHWAIT_TYPHOON   ← BERTHWAIT_TYPHOON  x 24 x 365 ÷ SIMULATIONTIME
  BERTHDELAY_STRIKE   ← BERTHDELAY_STRIKE  x 24 x 365 ÷ SIMULATIONTIME
  BERTHDELAY_TYPHOON  ← BERTHDELAY_TYPHOON x 24 x 365 ÷ SIMULATIONTIME
  BERTHDELAY_LEAVING  ← BERTHDELAY_LEAVING x 24 x 365 ÷ SIMULATIONTIME
  BERTHDELAY_SHIFTING ← BERTHDELAY_SHIFTING   x24x365 ÷ SIMULATIONTIME
  BERTHDELAY_BREAKDOWN ← BERTHDELAY_BREAKDOWN x24x365 ÷ SIMULATIONTIME
  WRITE " " TO REPORT WITH IMAGE A
  WRITE "PERFORMANCE OF BERTH";BERTH_NUMBER TO REPORT WITH IMAGE A^xx
  IF BERTH_SINGLE = TRUE
    WRITE "THIS BERTH IS A SINGLE BERTH" TO REPORT WITH IMAGE A
    WRITE "ANNUAL OCCUPATION-RATE =";BERTH_OCC_TIME ÷ 365 x 24 TO
    REPORT WITH IMAGE A^x.xxx IF BERTH_SINGLE = TRUE
    WRITE "ANNUAL OCCUPATIONTIME (HOURS):" TO REPORT WITH IMAGE A
    WRITE "   A) IN OPERATION  (NET OPERATION TIME)         =";
    BERTH_WORK_TIME - BERTHDELAY_BREAKDOWN TO REPORT WITH
    IMAGE A^xxxxxxx
    WRITE "   B) IN PARTIAL OPERATION (BREAKDOWN OF CRANE)  =";
```

```
          BERTHDELAY_BREAKDOWN TO REPORT WITH IMAGE A^xxxxxxx
 92       WRITE "   C) NOT IN OPERATION DUE TO DELAYS            =";
          BERTH_OCC_TIME - BERTH_WORK_TIME TO REPORT WITH IMAGE A^xxxxxxx
 93       WRITE "   TOTAL                                        =       ";
          BERTH_OCC_TIME TO REPORT WITH IMAGE A^xxxxxxx
 94       WRITE "ANNUAL TIME NOT OCCUPIED (HOURS)                =       ";
          (24 x 365) - BERTH_OCC_TIME TO REPORT WITH IMAGE A^xxxxxxx
 95     END
 96     IF BERTH_SINGLE = FALSE
 97       WRITE "THIS BERTH IS A MULTIPLE BERTH" TO REPORT WITH IMAGE A
 98       WRITE "ANNUAL OCCUPATION-RATE =";BERTH_OCC_TIME ÷ BERTH_LENGTH x
          365 x 24 TO REPORT WITH IMAGE A^x.xxx
 99       WRITE "ANNUAL OCCUPATIONTIME (METER-DAYS):" TO REPORT WITH
          IMAGE A
100       WRITE "   A) IN OPERATION                              =";
          (BERTH_WORK_TIME - BERTHDELAY_BREAKDOWN) ÷ 24 TO REPORT WITH
          IMAGE A^xxxxxxx
101       WRITE "   B) IN PARTIAL OPERATION (BREAKDOWN OF CRANE) =";
          BERTHDELAY_BREAKDOWN ÷ 24 TO REPORT WITH IMAGE A^xxxxxxx
102       WRITE "   C) NOT IN OPERATION DUE TO DELAYS            =";
          (BERTH_OCC_TIME - BERTH_WORK_TIME) ÷ 24 TO REPORT WITH
          IMAGE A^xxxxxxx
103       WRITE "   TOTAL                                        =       ";
          BERTH_OCC_TIME ÷ 24 TO REPORT WITH IMAGE A^xxxxxxx
104       WRITE "ANNUAL TIME NOT OCCUPIED (METER-DAYS)           =       ";
          (365 x BERTH_LENGTH) - BERTH_OCC_TIME ÷ 24 TO REPORT WITH
          IMAGE A^xxxxxxx
105       WRITE "(TOTAL ANNUAL AVAILABLE METER-DAYS  = 365 x";BERTH_LENGTH;
           "=       ";365xBERTH_LENGTH;")" TO REPORT WITH
          IMAGE A^xxxx^A^xxxxxxxA
106       WRITE " " TO REPORT WITH IMAGE A
107       WRITE "DISTR. OF NUMBER OF SHIPS AT THIS BERTH:";
          "SEE STORESTREAM Q_BERTH";BERTH_NUMBER TO REPORT WITH IMAGE A^Axx
108       WRITE "ANNUAL AVERAGE NUMBER OF SHIFTINGS OF SHIPS =";
          BERTH_SHIP_SHIFTS x 365÷SIMULATIONTIME TO REPORT WITH IMAGE A^xxx
109       WRITE "ANNUAL AVERAGE NUMBER OF SHIFTINGS OF CRANES =";
          BERTH_CRANE_SHIFTS x 365÷SIMULATIONTIME TO REPORT WITH IMAGE A^xxx
110     END
111     WRITE " " TO REPORT WITH IMAGE A
112     WRITE "PERFORMANCE OF CRANES AT THIS BERTH" TO REPORT WITH IMAGE A
113     FOR EACH CRANE IN BERTH_CRANES
114       CRANE_WORKTIME_B ← 0 IF ABS(CRANE_WORKTIME_B) < 0.005
115       CRANE_DOWNTIME   ← 0 IF ABS(CRANE_DOWNTIME) < 0.005
116       CRANE_RESTTIME   ← 0 IF ABS(CRANE_RESTTIME) < 0.005
```

```
        CRANE_DELAYTIME  ← 0 IF ABS(CRANE_DELAYTIME) < 0.005
        WRITE NAME OF THIS CRANE;"; ANNUAL TIME SPENT (HOURS):" TO REPORT
        WITH IMAGE A^A
        WRITE "   A) IN OPERATION                              =";24 x
        (365 ÷ SIMULATIONTIME) x CRANE_WORKTIME_B TO REPORT WITH
        IMAGE A^xxxxxxx
        WRITE "   B) NOT IN OPERATION DUE TO A BREAKDOWN       =";24 x
        (365 ÷ SIMULATIONTIME) x CRANE_DOWNTIME TO REPORT WITH
        IMAGE A^xxxxxxx
        WRITE "   C) NOT IN OPERATION DUE TO DELAYS            =";24 x
        (365 ÷ SIMULATIONTIME) x CRANE_DELAYTIME TO REPORT WITH
        IMAGE A^xxxxxxx
        WRITE "   D) IN MAINTENANCE                            =";24 x
        (365 ÷ SIMULATIONTIME) x CRANE_WORKTIME_B x MAINTENANCE_DURATION ÷
        MAINTENANCE_TIME TO REPORT WITH IMAGE A^xxxxxxx
        WRITE "   E) AT REST                                   =";24 x
        (365 ÷ SIMULATIONTIME) x ((CRANE_RESTTIME - LENGTH_ARRIVALPATT) -
        (CRANE_WORKTIME_B x MAINTENANCE_DURATION ÷ MAINTENANCE_TIME)) TO
        REPORT WITH IMAGE A^xxxxxxx
      END
    END
    WRITE " " TO REPORT WITH IMAGE A
    SIMULATIONTIME ← SIMULATIONTIME + LENGTH_ARRIVALPATT
  END

  @  Ship performances  @
  WRITE " " TO REPORT WITH IMAGE A
  WRITE "########## PERFORMANCE OF SHIPS ##########" TO REPORT WITH
  IMAGE A
  WRITE " " TO REPORT WITH IMAGE A
  I ← 0
  FOR EACH SHIPCLASS IN CLASSES
    CLASSDELAY_SHIFTING ← 0 IF ABS(CLASSDELAY_SHIFTING) < 0.005
    CLASSDELAY_TYPHOON  ← 0 IF ABS(CLASSDELAY_TYPHOON)  < 0.005
    CLASSDELAY_LEAVING  ← 0 IF ABS(CLASSDELAY_LEAVING)  < 0.005
    CLASSDELAY_STRIKE   ← 0 IF ABS(CLASSDELAY_STRIKE)   < 0.005
    CLASSDELAY_TIDE     ← 0 IF ABS(CLASSDELAY_TIDE)     < 0.005
    CLASSWAIT_TIDE      ← 0 IF ABS(CLASSWAIT_TIDE)      < 0.005
    CLASSWAIT_TYPHOON   ← 0 IF ABS(CLASSWAIT_TYPHOON)   < 0.005
    I ← I + 1
    WRITE "---------------------" TO REPORT WITH IMAGE A
    WRITE "CLASS";CLASS_NUMBER;":";NAME OF THIS SHIPCLASS TO REPORT WITH
    IMAGE A^xx^A^A
    WRITE "---------------------" TO REPORT WITH IMAGE A
  WRITE " " TO REPORT WITH IMAGE A
  WRITE "ANNUAL NUMBER OF SHIPS =";CLASS_ALLSHIPS x 365 ÷ SIMULATIONTIME
  - LENGTH_ARRIVALPATT TO REPORT WITH IMAGE A^xxxxx.xx
  WRITE "AVERAGE RATIO (TIME AT ANCHORAGE\TIME AT QUAY) =";
  CLASS_WAITINGTIME ÷ CLASS_BERTHTIME TO REPORT WITH IMAGE A^xx.xxx
  WRITE " " TO REPORT WITH IMAGE A
  WRITE "AVERAGE TIME AT ANCHORAGE (HOURS)       =        ";24 x
  CLASS_WAITINGTIME ÷ CLASS_ALLSHIPS TO REPORT WITH IMAGE A^xxxx.xx
  WRITE "AVERAGE MOORING TIME (HOURS)            =        ";24 x
  CLASS_MOORINGTIME ÷ CLASS_ALLSHIPS TO REPORT WITH IMAGE A^xxxx.xx
  WRITE "AVERAGE TIME AT QUAY (HOURS):" TO REPORT WITH IMAGE A
  WRITE "   A)  IN OPERATION                     =";24 x
  (CLASS_WORK_TIME - CLASSDELAY_BREAKDOWN) ÷ CLASS_ALLSHIPS TO REPORT
  WITH IMAGE A^xxxx.xx
  WRITE "   B)  IN PARTIAL OPERATION             =";24 x
  CLASSDELAY_BREAKDOWN ÷ CLASS_ALLSHIPS TO REPORT WITH IMAGE A^xxxx.xx
  WRITE "   C)  NOT IN OPERATION DUE TO DELAYS   =";24 x
  (CLASS_BERTHTIME - CLASS_WORK_TIME) ÷ CLASS_ALLSHIPS TO REPORT WITH
  IMAGE A^xxxx.xx
  WRITE "   TOTAL                                =        ";24 x
  CLASS_BERTHTIME ÷ CLASS_ALLSHIPS TO REPORT WITH IMAGE A^xxxx.xx
  WRITE " " TO REPORT WITH IMAGE A
  WRITE " " TO REPORT WITH IMAGE A
END

WRITE "########## DETAILED PERFORMANCE OF THE BERTHS ##########" TO
REPORT WITH IMAGE A
WRITE " " TO REPORT WITH IMAGE A
SIMULATIONTIME ← SIMULATIONTIME - LENGTH_ARRIVALPATT
@  Detailed performance of berths at this terminal  @
FOR EACH TERMINAL IN TERMINALSET
 FOR EACH BERTH IN TERM_BERTHSET
  IF BERTH_SINGLE = TRUE
   WRITE "BERTH";BERTH_NUMBER;"(SINGLE BERTH)" TO REPORT WITH IMAGE
   A^xx^A
   WRITE "ANNUAL OCCUPATIONTIME (HOURS):" TO REPORT WITH IMAGE A
   WRITE "   A) IN OPERATION  (NET OPERATION TIME)        =";
   BERTH_WORK_TIME - BERTHDELAY_BREAKDOWN TO REPORT WITH IMAGE A^xxxxxxx
   WRITE "   B) IN PARTIAL OPERATION (BREAKDOWN OF CRANE) =";
   BERTHDELAY_BREAKDOWN TO REPORT WITH IMAGE A^xxxxxxx
   WRITE "   C) NOT IN OPERATION:" TO REPORT WITH IMAGE A
   WRITE "       1) GROSS - NET OPERATION TIME            =";
   BERTH_OCC_TIME - BERTHDELAY_LEAVING + BERTHDELAY_TYPHOON +
   BERTH_WORK_TIME + BERTHDELAY_STRIKE TO REPORT WITH IMAGE A^xxxxxxx
   WRITE "      2) DUE TO STRIKE                               =";
   BERTHDELAY_STRIKE TO REPORT WITH IMAGE A^xxxxxxx
   WRITE "      3) DUE TO BAD WEATHER                          =";
   BERTHDELAY_TYPHOON TO REPORT WITH IMAGE A^xxxxxxx
   WRITE "      4) DUE TO WAITING WHEN LEAVING                 =";
   BERTHDELAY_LEAVING TO REPORT WITH IMAGE A^xxxxxxx
   WRITE "ANNUAL TIME NOT OCCUPIED (HOURS):" TO REPORT WITH IMAGE A
   WRITE "   A) DUE TO BAD WEATHER                             =";
   BERTHWAIT_TYPHOON TO REPORT WITH IMAGE A^xxxxxxx
   WRITE "   B) DUE TO TIDE                                    =";
   BERTHWAIT_TIDE TO REPORT WITH IMAGE A^xxxxxxx
   WRITE "   C) NO SHIPS FOR THIS BERTH AT ANCHORAGE           =";
   (24 x 365) - BERTH_OCC_TIME + BERTHWAIT_TYPHOON + BERTHWAIT_TIDE
   TO REPORT WITH IMAGE A^xxxxxxx
   WRITE " " TO REPORT WITH IMAGE A
  END
  IF BERTH_SINGLE = FALSE
   WRITE "BERTH";BERTH_NUMBER;"(MULTIPLE BERTH)" TO REPORT WITH IMAGE
   A^xx^A
   WRITE "ANNUAL OCCUPATIONTIME (METER-DAYS):" TO REPORT WITH IMAGE A
   WRITE "   A) IN OPERATION  (NET OPERATION TIME)             =";
   (BERTH_WORK_TIME - BERTHDELAY_BREAKDOWN) ÷ 24 TO REPORT WITH
   IMAGE A^xxxxxxx
   WRITE "   B) IN PARTIAL OPERATION (BREAKDOWN OF CRANE) =";
   BERTHDELAY_BREAKDOWN ÷ 24 TO REPORT WITH IMAGE A^xxxxxxx
   WRITE "   C) NOT IN OPERATION:" TO REPORT WITH IMAGE A
   WRITE "      1) GROSS - NET OPERATION TIME                  =";
   (BERTH_OCC_TIME - BERTHDELAY_STRIKE + BERTH_WORK_TIME +
   BERTHDELAY_TYPHOON + BERTHDELAY_SHIFTING + BERTHDELAY_LEAVING) ÷ 24
   TO REPORT WITH IMAGE A^xxxxxxx
   WRITE "      2) DUE TO STRIKE                               =";
   BERTHDELAY_STRIKE ÷ 24 TO REPORT WITH IMAGE A^xxxxxxx
   WRITE "      3) DUE TO BAD WEATHER                          =";
   BERTHDELAY_TYPHOON ÷ 24 TO REPORT WITH IMAGE A^xxxxxxx
   WRITE "      4) DUE TO WAITING WHEN LEAVING                 =";
   BERTHDELAY_LEAVING ÷ 24 TO REPORT WITH IMAGE A^xxxxxxx
   WRITE "      5) DUE TO SHIFTING OF SHIPS                    =";
   BERTHDELAY_SHIFTING ÷ 24 TO REPORT WITH IMAGE A^xxxxxxx
   WRITE "ANNUAL TIME NOT OCCUPIED (METER-DAYS):" TO REPORT WITH IMAGE A
   WRITE "   A) DUE TO BAD WEATHER                             =";
   BERTHWAIT_TYPHOON ÷ 24 TO REPORT WITH IMAGE A^xxxxxxx
   WRITE "   B) DUE TO TIDE                                    =";
   BERTHWAIT_TIDE ÷ 24 TO REPORT WITH IMAGE A^xxxxxxx
   WRITE "   C) NO SHIPS FOR THIS BERTH AT ANCHORAGE           =";
```

```
      (365 x BERTH_LENGTH) - (BERTH_OCC_TIME + BERTHWAIT_TYPHOON +
      BERTHWAIT_TIDE) ÷ 24 TO REPORT WITH IMAGE A^xxxxxxx
      WRITE " " TO REPORT WITH IMAGE A
     END
   END
 END
 WRITE " " TO REPORT WITH IMAGE A

 @  Indication for performance of terminal-equipment  @
 WRITE "####### INDICATION FOR PERFORMANCE OF TERMINAL-EQUIPMENT ######"
 TO REPORT  WITH IMAGE A
 WRITE " " TO REPORT WITH IMAGE A
 WRITE "INDICATION OF ANNUAL AVERAGE NUMBER OF HOURS IN OPERATION" TO
 REPORT WITH IMAGE A
 WRITE " " TO REPORT WITH IMAGE A
 FOR EACH TERMINAL IN TERMINALSET
    TERM_NET_FAC ← 0    @  Alternative use of this attribute  @
    FOR EACH CARGOTYPE IN TERM_CARGO
       TERM_NET_FAC ← TERM_NET_FAC + 10 x (CT_THROUGHPUT_EXP +
       CT_THROUGHPUT_IMP) x 365 ÷ SIMULATIONTIME - 100 IF CT_CODE = "A"
       TERM_NET_FAC ← TERM_NET_FAC + (CT_THROUGHPUT_EXP +
       CT_THROUGHPUT_IMP) x 365 ÷ SIMULATIONTIME - 100 IF CT_CODE ≠ "A"
    END
    WRITE NAME OF THIS TERMINAL;":";(TERM_EQUIP_PERC÷100)xTERM_NET_FAC ÷
    (TERM_EQUIP_UNITS x TERM_EQUIP_CAP);"HOURS" TO REPORT WITH
    IMAGE A^A^xxxx^A
    WRITE " " TO REPORT WITH IMAGE A
 END
 WRITE " " TO REPORT WITH IMAGE A

 @  Detailed performances of the ships  @
 WRITE "########## DETAILED PERFORMANCE OF SHIPS ##########" TO REPORT
 WITH IMAGE A
 WRITE " " TO REPORT WITH IMAGE A
 I ← 0
 FOR EACH SHIPCLASS IN CLASSES
   I ← I + 1
   WRITE "CLASS";CLASS_NUMBER;":";NAME OF THIS SHIPCLASS TO REPORT WITH
   IMAGE A^xx^A^A
   WRITE "AVERAGE TIME AT ANCHORAGE (HOURS):" TO REPORT WITH IMAGE A
   WRITE "    A)  WAITING DUE TO FULL PORT              =";24 x
   (CLASS_WAITINGTIME - CLASSWAIT_TIDE + CLASSWAIT_TYPHOON) ÷
   CLASS_ALLSHIPS TO REPORT WITH  IMAGE A^xxxx.xx
   WRITE "    B)  WAITING DUE TO BAD WEATHER            =";24 x
```

```
    CLASSWAIT_TYPHOON ÷ CLASS_ALLSHIPS TO REPORT WITH IMAGE A^xxxx.xx
    WRITE "   C)  WAITING DUE TO TIDE                             =";24 x
    CLASSWAIT_TIDE ÷ CLASS_ALLSHIPS TO REPORT WITH IMAGE A^xxxx.xx
    WRITE "AVERAGE MOORING TIME (HOURS)                           =";24 x
    CLASS_MOORINGTIME ÷ CLASS_ALLSHIPS TO REPORT WITH IMAGE A^xxxx.xx
    WRITE "AVERAGE TIME AT QUAY (HOURS):" TO REPORT WITH IMAGE A
    WRITE "   A)  IN OPERATION (NET OPERATION TIME)               =";24 x
    (CLASS_WORK_TIME - CLASSDELAY_BREAKDOWN) ÷ CLASS_ALLSHIPS TO REPORT
    WITH IMAGE A^xxxx.xx
    WRITE "   B)  IN PARTIAL OPERATION (CRANE-BREAKDOWN) =";24 x
    CLASSDELAY_BREAKDOWN ÷ CLASS_ALLSHIPS TO REPORT WITH IMAGE A^xxxx.xx
    WRITE "   C)  NOT IN OPERATION:" TO REPORT WITH IMAGE A
    WRITE "        1)  GROSS - NET OPERATION TIME                 =";24 x
    (CLASS_BERTHTIME - CLASSDELAY_TYPHOON + CLASSDELAY_STRIKE +
    CLASSDELAY_SHIFTING + CLASS_WORK_TIME + CLASSDELAY_LEAVING) ÷
    CLASS_ALLSHIPS TO REPORT WITH IMAGE A^xxxx.xx
    WRITE "        2)  WAITING DUE TO STRIKES                     =";24 x
    CLASSDELAY_STRIKE ÷ CLASS_ALLSHIPS TO REPORT WITH IMAGE A^xxxx.xx
    WRITE "        3)  WAITING DUE TO BAD WEATHER                 =";24 x
    CLASSDELAY_TYPHOON ÷ CLASS_ALLSHIPS TO REPORT WITH IMAGE A^xxxx.xx
    WRITE "        4)  WAITING WHEN LEAVING BERTH                 =";24 x
    CLASSDELAY_LEAVING ÷ CLASS_ALLSHIPS TO REPORT WITH IMAGE A^xxxx.xx
    WRITE "        5)  WAITING DUE TO SHIFTING                    =";24 x
    CLASSDELAY_SHIFTING ÷ CLASS_ALLSHIPS TO REPORT WITH IMAGE A^xxxx.xx
    WRITE "NB. AVERAGE WAITING IN PORT DUE TO TIDE                =";24 x
    CLASSDELAY_TIDE ÷ CLASS_ALLSHIPS TO REPORT WITH IMAGE A^xxxx.xx
    WRITE " " TO REPORT WITH IMAGE A
  END

  @  Restrictions data  @
  WRITE " " TO REPORT WITH IMAGE A
  WRITE "AVERAGE ANNUAL NUMBER OF TYPHOONS IN THIS PORT =";NR_TYPHOONS x
  365 ÷ SIMULATIONTIME TO REPORT WITH IMAGE A^xx.xx
  WRITE "AVERAGE ANNUAL NUMBER OF STRIKES IN THIS PORT  =";NR_STRIKES x
  365 ÷ SIMULATIONTIME TO REPORT WITH IMAGE A^xx.xx
  WRITE " " TO REPORT WITH IMAGE A

  PASSIVATE
```

```
MODEL HP2_11P                                          Date:  93/08/16
MOD PASSENGERSHIPMOD                                   Time:  18:25:47

@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@  PROCESS OF A PASSENGER SHIP  @
@  for pontianak                @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

PASSENGERSHIPSTART:
   WAIT LENGTH_ARRIVALPATT
   ENTER ROW
   ENTER PRIORITY_ROW IF CLASS_PRIORITY OF RIGHTCLASS = TRUE
   ENTER TERM_SHIPS OF RIGHTTERM
   TERM_ALLSHIPS OF RIGHTTERM ← TERM_ALLSHIPS OF RIGHTTERM + 1
   PASSIVATE
   SHIPCALL1 ← NOW
   WAIT WHILE TYPHOON_ALARM = TRUE
   THIS BERTH        ← RIGHTBERTH
   THIS SHIPCLASS    ← RIGHTCLASS
   CLASSWAIT_TYPHOON ← CLASSWAIT_TYPHOON + NOW - SHIPCALL1
   BERTHWAIT_TYPHOON ← BERTHWAIT_TYPHOON + NOW - SHIPCALL1
   IF BERTH_SINGLE = TRUE
   BERTHWAIT_TYPHOON ← BERTHWAIT_TYPHOON + SHIP_LENGTH_BERTHED x
   NOW - SHIPCALL1 IF BERTH_SINGLE = FALSE
   LEAVE ROW
   LEAVE PRIORITY_ROW IF CLASS_PRIORITY OF RIGHTCLASS = TRUE

   ENTER PORT
   WAIT MAX(0,SHIP_MOORINGTIME - 0.2) HOURS
   WAIT WHILE (MOORING[BERTH_NUMBER OF RIGHTBERTH + 2] = TRUE) v
   (MOORING[BERTH_NUMBER OF RIGHTBERTH + 1] = TRUE) v
   (MOORING[BERTH_NUMBER OF RIGHTBERTH] = TRUE)
   SHIP_MOORING_EXTRA ← NOW - QUEUETIME IN PORT +
   MAX(0,(SHIP_MOORINGTIME-0.2)÷24)
   MOORING[BERTH_NUMBER OF RIGHTBERTH + 1] ← TRUE
   WAIT 0.2 HOURS
   MOORING[BERTH_NUMBER OF RIGHTBERTH + 1] ← FALSE
   SHIPCALL1 ← NOW
   SHIP_CRANE_ALLOC ← TRUE
   ENTER QUAY
   SHIPCALL1 ← NOW
   WAIT 5 ÷ 24
   LEAVE BERTH_SHIPS OF RIGHTBERTH
   LEAVE QUAY
```

```
    SHIPCALL2 ← NOW
SHIPDELAY2:
    SHIPCALL1 ← NOW
    WAIT WHILE TYPHOON_ALARM = TRUE
    THIS BERTH           ← RIGHTBERTH
    THIS SHIPCLASS       ← RIGHTCLASS
    CLASSDELAY_TYPHOON ← CLASSDELAY_TYPHOON + NOW - SHIPCALL1
    BERTHDELAY_TYPHOON ← BERTHDELAY_TYPHOON + NOW - SHIPCALL1
    IF BERTH_SINGLE = TRUE
    BERTHDELAY_TYPHOON ← BERTHDELAY_TYPHOON + SHIP_LENGTH_BERTHED x
    NOW - SHIPCALL1 IF BERTH_SINGLE = FALSE
    SHIPCALL1 ← NOW
    WAIT WHILE (MOORING[BERTH_NUMBER OF RIGHTBERTH + 2] = TRUE) v
    (MOORING[BERTH_NUMBER OF RIGHTBERTH + 1] = TRUE) v
    (MOORING[BERTH_NUMBER OF RIGHTBERTH] = TRUE)
    CLASSDELAY_LEAVING ← CLASSDELAY_LEAVING + NOW - SHIPCALL1
    BERTHDELAY_LEAVING ← BERTHDELAY_LEAVING + NOW - SHIPCALL1
    IF BERTH_SINGLE = TRUE
    BERTHDELAY_LEAVING ← BERTHDELAY_LEAVING + SHIP_LENGTH_BERTHED x
    NOW - SHIPCALL1 IF BERTH_SINGLE = FALSE
    REPEAT FROM SHIPDELAY2 IF TYPHOON_ALARM = TRUE

    THIS BERTH          ← RIGHTBERTH
    THIS SHIPCLASS      ← RIGHTCLASS
    BERTH_LENGTH_FREE ← BERTH_LENGTH_FREE + SHIP_LENGTH_BERTHED
    CLASS_ALLSHIPS      ← CLASS_ALLSHIPS + 1
    BERTH_OCC_TIME      ← BERTH_OCC_TIME + NOW - QUEUETIME IN PORT +
    (SHIP_MOORINGTIME ÷ 24) + SHIP_MOORING_EXTRA IF BERTH_SINGLE = TRUE
    BERTH_OCC_TIME      ← BERTH_OCC_TIME + SHIP_LENGTH_BERTHED x NOW -
    QUEUETIME IN PORT + (SHIP_MOORINGTIME ÷ 24) + SHIP_MOORING_EXTRA
    IF BERTH_SINGLE = FALSE
    CLASS_BERTHTIME     ← CLASS_BERTHTIME + NOW - QUEUETIME IN PORT +
    (SHIP_MOORINGTIME ÷ 24) + SHIP_MOORING_EXTRA
    CLASS_WAITINGTIME ← CLASS_WAITINGTIME + QUEUETIME IN PORT - QUEUETIME
    IN TERM_SHIPS OF RIGHTTERM
    CLASS_MOORINGTIME ← CLASS_MOORINGTIME + (SHIP_MOORINGTIME ÷ 24) +
    SHIP_MOORING_EXTRA
    LEAVE TERM_SHIPS OF RIGHTTERM
    MOORING[BERTH_NUMBER OF RIGHTBERTH + 1] ← TRUE
    WAIT 0.2 HOURS
    MOORING[BERTH_NUMBER OF RIGHTBERTH + 1] ← FALSE
    LEAVE PORT
    WAIT MAX(0,SHIP_MOORINGTIME - 0.2) HOURS
    TERMINATE
```

MODEL HP2_11P   Date: 93/08/16
MAC PATTERNS   Time: 18:33:58

```
@ -------------------------------------------------- @
@  MACRO PATTERNS                                      @
@  Create general arrival- and departure-pattern      @
@ -------------------------------------------------- @

L ← 0
FOR J ← 1 TO (LENGTH_ARRIVALPATT - SATSUN)
   THIS ARRIVALDAY ← NEW ARRIVALDAY
   ARRDAY_PERC ← READ FROM TFILE
   ARRDAY_NUM  ← J
   JOIN THIS ARRIVALDAY TO CT_ARRIVALPATT
   L ← L + ARRDAY_PERC
END
CT_PERC_ARR_DIRECT ← READ FROM TFILE
L ← L + CT_PERC_ARR_DIRECT
CALL ERROR2("DAY-ARR",CT_NUMBER) IF L ▬ 100
CT_PERC_ARR_ROAD    ← READ FROM TFILE
CT_PERC_ARR_RAIL    ← READ FROM TFILE
CT_PERC_ARR_IWT     ← READ FROM TFILE
CALL ERROR2("MODE-ARR",CT_NUMBER) IF (CT_PERC_ARR_ROAD + CT_PERC_ARR_RAIL
+ CT_PERC_ARR_IWT) ▬ 100

CT_PERC_DEP_DIRECT ← READ FROM TFILE
L ← CT_PERC_DEP_DIRECT
FOR J ← 1 TO LENGTH_DEPARTUREPATT
   THIS DEPARTUREDAY ← NEW DEPARTUREDAY
   DEPDAY_PERC ← READ FROM TFILE
   DEPDAY_NUM  ← J
   JOIN THIS DEPARTUREDAY TO CT_DEPARTUREPATT
   L ← L + DEPDAY_PERC
END
CALL ERROR2("DAY-DEP",CT_NUMBER) IF L ▬ 100
CT_PERC_DEP_ROAD    ← READ FROM TFILE
CT_PERC_DEP_RAIL    ← READ FROM TFILE
CT_PERC_DEP_IWT     ← READ FROM TFILE
CALL ERROR2("MODE-DEP",CT_NUMBER) IF (CT_PERC_DEP_ROAD + CT_PERC_DEP_RAIL
+ CT_PERC_DEP_IWT) ▬ 100

RETURN
```

```
MODEL HP2_11P                                                    Date:  93/08/16
MAC ARRIVALPATTERN                                               Time:  18:34:00

@ ------------------------------------------------------ @
@  MACRO ARRIVALPATTERN                                  @
@  Check inland transport arrival pattern and reshuffle  @
@  in case of not working on saturdays and sundays       @
@ ------------------------------------------------------ @

THIS TERMINAL  ← RIGHTTERM
THIS CARGOTYPE ← FIRST CARGOTYPE IN TERM_CARGO WITH CT_NUMBER =
SHIP_CARGOTYPE

J ← LENGTH_ARRIVALPATT - SATSUN
FOR H ← 1 TO LENGTH_ARRIVALPATT
   I ← LENGTH_ARRIVALPATT + 1 - H
   IF (((DAYNUMBER + I - 7 x FLOOR((DAYNUMBER + I) ÷ 7)) = 6) ^
       (TERM_SAT_TRANS = FALSE)) v
       (((DAYNUMBER + I - 7 x FLOOR((DAYNUMBER + I) ÷ 7)) = 0) ^
       (TERM_SUN_TRANS = FALSE)) v (J = 0)
      SHIP_PERC_ARR[I] ← 0
      GOTO NEXT_ARRDAY
   END
   THIS ARRIVALDAY  ← FIRST ARRIVALDAY IN CT_ARRIVALPATT WITH
                      ARRDAY_NUM = J
   SHIP_PERC_ARR[I] ← ARRDAY_PERC
   J ← J - 1
   NEXT_ARRDAY:
END

RETURN
```

```
MODEL HP2_11P                                             Date:  93/08/16
MAC DEPARTUREPATTERN                                      Time:  18:34:05

@ ------------------------------------------------------------- @
@  MACRO DEPARTUREPATTERN                                        @
@  Check inland transport departure pattern and count extra days @
@  in case of not working on saturdays and sundays               @
@ ------------------------------------------------------------- @

THIS TERMINAL  ← RIGHTTERM
THIS CARGOTYPE ← FIRST CARGOTYPE IN TERM_CARGO WITH CT_NUMBER =
SHIP_CARGOTYPE
FOR I ← 1 TO LENGTH_DEPARTUREPATT
   THIS DEPARTUREDAY ← FIRST DEPARTUREDAY IN CT_DEPARTUREPATT WITH
   DEPDAY_NUM = I
   SHIP_PERC_DEP[I] ← DEPDAY_PERC
END
FOR I ← 1 TO (LENGTH_DEPARTUREPATT + SHIP_EXTRA_DEP)
   IF (((DAYNUMBER + I - 7 x FLOOR((DAYNUMBER + I) ÷ 7)) = 6) ∧
       (TERM_SAT_TRANS = FALSE)) v
       (((DAYNUMBER + I - 7 x FLOOR((DAYNUMBER + I) ÷ 7)) = 0) ∧
       (TERM_SUN_TRANS = FALSE))
       SHIP_EXTRA_DEP ← SHIP_EXTRA_DEP + 1
   END
END

RETURN
```

```
MODEL HP2_11P                                                   Date:  93/08/16
MAC PATTERNCHANGE                                               Time:  18:34:11

@ ------------------------------------------------------- @
@  MACRO PATTERNCHANGE                                     @
@  Change arrivalpattern of ships in queue "arrivingships" @
@  in case of strikes or typhoons                          @
@ ------------------------------------------------------- @

PARAMETER:
REAL:  SPELL

IF SPELL < LENGTH_ARRIVALPATT + SHIP_EXTRA_ARR - SHIP_DAY
   FOR I ← 1 TO SPELL
      SHIP_PERC_ARR[SHIP_DAY + SPELL + 1] ←
      SHIP_PERC_ARR[SHIP_DAY + SPELL + 1] +
      SHIP_PERC_ARR[SHIP_DAY + I]
      SHIP_PERC_ARR[SHIP_DAY + I] ← 0
   END
   SHIP_DAY ← SHIP_DAY + SPELL
   RETURN
END
IF SPELL ≥ LENGTH_ARRIVALPATT + SHIP_EXTRA_ARR - SHIP_DAY
   J ← 0
   FOR I ← (SHIP_DAY + 1) TO (LENGTH_ARRIVALPATT + SHIP_EXTRA_ARR)
      J ← J + SHIP_PERC_ARR[I]
   SHIP_PERC_ARR[I] ← 0
   END
   SHIP_PERC_ARR[SHIP_DAY + 1] ← J
   SHIP_EXTRA_ARR ← MAX(((SPELL+1) - (LENGTH_ARRIVALPATT) - (FLOOR(NOW) -
   FLOOR(QUEUETIME OF THIS SHIP IN ARRIVINGSHIPS))),0)
END
RETURN
```

```
MODEL HP2_11P                                            Date:  93/08/16
MAC ERROR1                                               Time:  18:34:17

@ --------------- @
@  MACRO ERROR1   @
@  Illegal entry  @
@ --------------- @

PARAMETER:
   CHARACTER(10) :  KARAKTER
   INTEGER       :  C

WRITE "ERROR:" TO CHECKLIST WITH IMAGE A
WRITE "ILLEGAL ENTRY 0 FOR";KARAKTER TO CHECKLIST WITH IMAGE A^A IF C = 1
WRITE "ILLEGAL ENTRY, NOT 0 OR 1 FOR";KARAKTER TO CHECKLIST WITH
IMAGE A^A IF C = 2
WRITE "ILLEGAL ENTRY, VALUE FOR";KARAKTER;"IS TOO SMALL" TO CHECKLIST WITH
 IMAGE A^A^A IF C = 3
WRITE "ILLEGAL ENTRY, VALUE FOR";KARAKTER;"IS TOO BIG" TO CHECKLIST WITH
IMAGE A^A^A IF C = 4
WRITE " " TO CHECKLIST WITH IMAGE A
ERRORS ← ERRORS + 1

RETURN
```

```
MODEL HP2_11P                                                    Date:  93/08/16
MAC ERROR2                                                       Time:  18:34:22

@ ---------------------------------- @
@  MACRO ERROR2                      @
@  Percentages do not add up to 100  @
@ ---------------------------------- @

PARAMETER:
   CHARACTER(10) : KARAKTER
   INTEGER       : NUM

WRITE "ERROR:" TO CHECKLIST WITH IMAGE A
WRITE KARAKTER;"PERCENTAGES OF ";NUM;"DO NOT ADD UP TO 100" TO CHECKLIST
WITH IMAGE AA^xxx^A
WRITE " " TO CHECKLIST WITH IMAGE A
ERRORS ← ERRORS + 1

RETURN
```

MODEL HP2_11P   Date: 93/08/16
MAC ERROR3   Time: 18:34:27

```
@ --------------- @
@   MACRO ERROR3   @
@   Data overflow  @
@ --------------- @

PARAMETER:
   CHARACTER(10) : KARAKTER

WRITE "ERROR:" TO CHECKLIST WITH IMAGE A
WRITE "DATA OVERFLOW IN";KARAKTER;"; CHECK THIS FILE" TO CHECKLIST WITH
 IMAGE A^A^A
WRITE " " TO CHECKLIST WITH IMAGE A
ERRORS ← ERRORS + 1

RETURN
```

```
MODEL HP2_11P                                                     Date:  93/08/16
MAC DWT_TABLES                                                    Time:  18:34:33

@ ------------------ @
@  MACRO DWT_TABLES  @
@ ------------------ @

FOR I ← 1 TO 4
   L ← READ FROM DFILE
   FOR J ← 1 TO L
      TAB_X ← READ FROM DFILE
      TAB_Y ← READ FROM DFILE
      TABULATE TAB_Y IN TAB_DRAUGHT[I] AT TAB_X
   END
   L ← READ FROM DFILE
   FOR J ← 1 TO L
      TAB_X ← READ FROM DFILE
      TAB_Y ← READ FROM DFILE
      TABULATE TAB_Y IN TAB_LENGTH[I] AT TAB_X
   END
END
```

MODEL HP2_11U   Date: 93/08/16
MOD HARBOURMOD   Time: 18:40:31

```
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@  PROCESS OF THE HARBOUR-MASTER  @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

HM_START:
  WAIT WHILE ROW IS EMPTY
  WAIT WHILE LENGTH OF PORT = PORT_CAP
  HM_CHECKSHIP ← FIRST SHIP IN ROW
  FOR L ← 1 TO LENGTH OF ROW
    THIS SHIP      ← HM_CHECKSHIP
    THIS SHIPCLASS ← RIGHTCLASS
    THIS TERMINAL  ← RIGHTTERM
    IF RIGHTBERTH IS NONE
      HM_CHECKBERTH ← FIRST BERTH IN TERM_BERTHSET
      FOR M ← 1 TO LENGTH OF TERM_BERTHSET
        THIS BERTH ← HM_CHECKBERTH
        HM_PRIORITY_SHIP ← FIRST SHIP IN PRIORITY_ROW WITH
        (RIGHTBERTH IS NONE) ^ (SHIP_TERM_NUMBER = BERTH_TERM_NUMBER) ^
        (CLASS_PRIORITY OF RIGHTCLASS = TRUE) ^
        (SHIP_DRAUGHT_BERTHED ≤ BERTH_DEPTH)
        IF (BERTH_DEPTH ≥ SHIP_DRAUGHT_BERTHED) ^
          (BERTH_LENGTH_FREE ≥ SHIP_LENGTH_BERTHED) ^
          (LENGTH OF BERTH_SHIPS < BERTH_CAP)
          IF HM_PRIORITY_SHIP IS NOT NONE
            FOR N ← 1 TO BERTH_CARGOTYPES
              GOTO HM_CONTINUE IF (HM_PRIORITY_SHIP IS NOT THIS SHIP) ^
              (BERTH_CARGOTYPE[N] = SHIP_CARGOTYPE OF HM_PRIORITY_SHIP)
            END
          END
          FOR N ← 1 TO BERTH_CARGOTYPES
            IF BERTH_CARGOTYPE[N] = SHIP_CARGOTYPE
              IF BERTH_SINGLE = FALSE
                SHIP_X_BOW ← 0 IF BERTH_SHIPS IS EMPTY
                CALL SHIFTSHIPS IF BERTH_SHIPS IS NOT EMPTY
                THIS SHIP ← HM_CHECKSHIP
                SHIP_X_STERN ← SHIP_X_BOW + SHIP_LENGTH_BERTHED
              END
              JOIN THIS SHIP TO BERTH_SHIPS RANKED BY SHIP_X_BOW
              BERTH_LENGTH_FREE ← BERTH_LENGTH_FREE - SHIP_LENGTH_BERTHED
              RIGHTBERTH ← THIS BERTH
              REACTIVATE THIS SHIP
            WAIT 0.25 ÷ 24
            REPEAT FROM HM_START
          END
        END
      END
      HM_CONTINUE:
      HM_PRIORITY_SHIP ← NONE
      HM_CHECKBERTH ← SUCC OF THIS BERTH IN TERM_BERTHSET
    END
  END
  HM_CHECKSHIP ← SUCC OF THIS SHIP IN ROW
END
HM_ROWLENGTH  ← LENGTH OF ROW
HM_PORTLENGTH ← LENGTH OF PORT
WAIT WHILE (LENGTH OF ROW=HM_ROWLENGTH) ^ (LENGTH OF PORT=HM_PORTLENGTH)
REPEAT FROM HM_START
```

MODEL HP2_11U   Date: 93/08/16
MAC SHIFTSHIPS   Time: 18:40:50

```
@ ----------------- @
@  MACRO SHIFTSHIPS  @
@ ----------------- @

SHIP_X_BOW OF HM_CHECKSHIP ← ~1
THIS BERTH    ← HM_CHECKBERTH
HM_SHIFTSHIP ← FIRST SHIP IN BERTH_SHIPS WITH SHIP_X_BOW > 0
WHILE SHIP_X_BOW OF HM_CHECKSHIP = ~1
  BERTH_SPACE_FREE ← BERTH_LENGTH
  IF SHIP_X_BOW OF FIRST OF BERTH_SHIPS ≥ SHIP_LENGTH_BERTHED OF
    HM_CHECKSHIP
    BERTH_SPACE_FREE ← SHIP_X_BOW OF FIRST OF BERTH_SHIPS
    SHIP_X_BOW OF HM_CHECKSHIP ← 0
  END
  FOR EACH SHIP IN BERTH_SHIPS
    IF SUCC OF THIS SHIP IN BERTH_SHIPS IS NOT NONE
      IF ((SHIP_X_BOW OF SUCC OF THIS SHIP IN BERTH_SHIPS - SHIP_X_STERN)
        ≥ SHIP_LENGTH_BERTHED OF HM_CHECKSHIP) ^ ((SHIP_X_BOW OF SUCC OF
        THIS SHIP IN BERTH_SHIPS - SHIP_X_STERN) < BERTH_SPACE_FREE)
        BERTH_SPACE_FREE ← SHIP_X_BOW OF SUCC OF THIS SHIP IN
        BERTH_SHIPS - SHIP_X_STERN
        SHIP_X_BOW OF HM_CHECKSHIP ← SHIP_X_STERN
      END
    END
  END
  IF ((BERTH_LENGTH - SHIP_X_STERN OF LAST OF BERTH_SHIPS) ≥
    SHIP_LENGTH_BERTHED OF HM_CHECKSHIP) ^ ((BERTH_LENGTH - SHIP_X_STERN
    OF LAST OF BERTH_SHIPS) < BERTH_SPACE_FREE)
    BERTH_SPACE_FREE ← BERTH_LENGTH - SHIP_X_STERN OF LAST OF BERTH_SHIPS
    SHIP_X_BOW OF HM_CHECKSHIP ← SHIP_X_STERN OF LAST OF BERTH_SHIPS
  END
  IF SHIP_X_BOW OF HM_CHECKSHIP = ~1
    SHIP_X_BOW OF HM_SHIFTSHIP ← 0 IF PRED OF HM_SHIFTSHIP IN
    BERTH_SHIPS IS NONE
    IF PRED OF HM_SHIFTSHIP IN BERTH_SHIPS IS NOT NONE
      SHIP_X_BOW OF HM_SHIFTSHIP ← SHIP_X_STERN OF PRED OF
      HM_SHIFTSHIP IN BERTH_SHIPS
    END
    SHIP_X_STERN OF HM_SHIFTSHIP ← SHIP_LENGTH_BERTHED OF
    HM_SHIFTSHIP + SHIP_X_BOW OF HM_SHIFTSHIP
    REMOVE HM_SHIFTSHIP FROM BERTH_SHIPS
```

```
33    JOIN HM_SHIFTSHIP TO BERTH_SHIPS RANKED BY SHIP_X_BOW OF HM_SHIFTSHIP
34    SHIP_SHIFTED OF HM_SHIFTSHIP ← TRUE
35    BERTH_SHIP_SHIFTS ← BERTH_SHIP_SHIFTS + 1
36    GOTO NEXTSHIFT IF RIGHTCRANE OF HM_SHIFTSHIP IS NONE
37    FOR EACH CRANE IN BERTH_CRANES WITH CRANE_MYSHIP IS HM_SHIFTSHIP
38      IF SUCC OF HM_SHIFTSHIP IN BERTH_SHIPS IS NOT NONE
39        CRANE_X_MAX ← SHIP_X_BOW OF SUCC OF HM_SHIFTSHIP IN BERTH_SHIPS
40      END
41      CRANE_X_MAX ← BERTH_LENGTH IF SUCC OF HM_SHIFTSHIP IN BERTH_SHIPS IS
        NONE
42      CRANE_X_MIN ← SHIP_X_BOW OF HM_SHIFTSHIP
43    END
44    THIS CRANE ← FIRST CRANE IN BERTH_CRANES WITH CRANE_X_MIN =
      SHIP_X_BOW OF HM_SHIFTSHIP
45    RIGHTCRANE OF HM_SHIFTSHIP ← THIS CRANE
46    SHIP_RATE OF HM_SHIFTSHIP ← CLASS_GEAR_UNLOADCAP OF RIGHTCLASS OF
      HM_SHIFTSHIP x 24 IF SHIP_UNLOADING OF HM_SHIFTSHIP = TRUE
47    SHIP_LOAD_RATE OF HM_SHIFTSHIP ← CLASS_GEAR_LOADCAP OF RIGHTCLASS OF
      HM_SHIFTSHIP x 24 IF SHIP_UNLOADING OF HM_SHIFTSHIP = TRUE
48    SHIP_RATE OF HM_SHIFTSHIP ← CLASS_GEAR_LOADCAP OF RIGHTCLASS OF
      HM_SHIFTSHIP x 24 IF SHIP_LOADING OF HM_SHIFTSHIP = TRUE
49    FOR I ← 1 TO SHIP_CRANE_SUPPLY OF HM_SHIFTSHIP
50      IF CRANE_AVAILABILITY = TRUE
51        CRANE_X_MIN ← SHIP_X_BOW OF HM_SHIFTSHIP
52        CRANE_X_MAX ← SHIP_X_STERN OF HM_SHIFTSHIP
53        SHIP_RATE OF HM_SHIFTSHIP ← SHIP_RATE OF HM_SHIFTSHIP +
          CRANE_UNLOADCAP x SHIP_FORTYFEETFAC_IMP OF HM_SHIFTSHIP IF
          SHIP_UNLOADING OF HM_SHIFTSHIP = TRUE
54        SHIP_LOAD_RATE OF HM_SHIFTSHIP ← SHIP_LOAD_RATE OF
          HM_SHIFTSHIP + CRANE_LOADCAP x SHIP_FORTYFEETFAC_EXP OF
          HM_SHIFTSHIP IF SHIP_UNLOADING OF HM_SHIFTSHIP = TRUE
55        SHIP_RATE OF HM_SHIFTSHIP ← SHIP_RATE OF HM_SHIFTSHIP +
          CRANE_LOADCAP x SHIP_FORTYFEETFAC_EXP OF HM_SHIFTSHIP IF
          SHIP_LOADING OF HM_SHIFTSHIP = TRUE
56        CRANE_MYSHIP ← HM_SHIFTSHIP
57        ACTIVATE THIS CRANE WITH DELAY ((1:24) + 0.00001) FROM CRANESTART
          IN CRANEMOD
58      END
59      IF CRANE_AVAILABILITY = FALSE
60        CRANE_PREV_SHIP ← CRANE_MYSHIP
61        CRANE_MYSHIP    ← HM_SHIFTSHIP
62      END
63      THIS CRANE ← SUCC OF THIS CRANE IN CRANESET
64      GOTO NEXTSHIFT IF THIS CRANE IS NONE
```

```
    END
    NEXTSHIFT:
    IF SUCC OF HM_SHIFTSHIP IN BERTH_SHIPS IS NOT NONE
      HM_SHIFTSHIP ← SUCC OF HM_SHIFTSHIP IN BERTH_SHIPS
    END
  END
END
FOR EACH CRANE IN BERTH_CRANES WITH CRANE_MYSHIP IS NONE
  THIS SHIP   ← FIRST SHIP IN BERTH_SHIPS WITH SHIP_X_BOW ≥ CRANE_X_MAX
  CRANE_X_MAX ← SHIP_X_BOW IF THIS SHIP IS NOT NONE
  CRANE_X_MAX ← BERTH_LENGTH IF THIS SHIP IS NONE
  THIS SHIP   ← LAST SHIP IN BERTH_SHIPS WITH SHIP_X_STERN ≤ CRANE_X_MIN
  CRANE_X_MIN ← SHIP_X_STERN IF THIS SHIP IS NOT NONE
  CRANE_X_MIN ← 0 IF THIS SHIP IS NONE
END
RETURN
```

```
MODEL HP2_11U                                                    Date:  93/08/16
MOD TERMINALMOD                                                  Time:  18:40:34

@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@  PROCESS OF THE TERMINAL-MASTER  @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

TM_START:
  TM_QUAYLENGTH ← LENGTH OF QUAY
  WAIT WHILE LENGTH OF QUAY = TM_QUAYLENGTH
  GOTO TM_REALLOCATE IF LENGTH OF QUAY < TM_QUAYLENGTH

  @  Crane-allocation (when a ship arrives) @
  FOR EACH SHIP IN QUAY WITH SHIP_CRANE_ALLOC = FALSE
    THIS BERTH       ← RIGHTBERTH
    THIS SHIPCLASS   ← RIGHTCLASS
    THIS TERMINAL    ← RIGHTTERM
    SHIP_UNLOAD_RATE ← 0
    SHIP_LOAD_RATE   ← 0
    @  Crane-allocation for a single berth  @
    IF BERTH_SINGLE = TRUE
      J ← 0
      THIS CRANE ← FIRST OF BERTH_CRANES
      WHILE (J < MIN(CLASS_CRANE_DEMAND,BERTH_CRANE_SUPPLY)) ^
        (J ≤ LENGTH OF BERTH_CRANES)
        IF CRANE_AVAILABILITY = TRUE
          SHIP_UNLOAD_RATE ← SHIP_UNLOAD_RATE + CRANE_UNLOADCAP x
                             SHIP_FORTYFEETFAC_IMP
          SHIP_LOAD_RATE   ← SHIP_LOAD_RATE + CRANE_LOADCAP x
                             SHIP_FORTYFEETFAC_EXP
          CRANE_MYSHIP     ← THIS SHIP
          ACTIVATE THIS CRANE FROM CRANESTART IN CRANEMOD
        END
        IF CRANE_AVAILABILITY = FALSE
          CRANE_PREV_SHIP ← CRANE_MYSHIP
          CRANE_MYSHIP    ← THIS SHIP
        END
        J ← J + 1
        THIS CRANE ← SUCC OF THIS CRANE IN BERTH_CRANES
        IF (J ≤ LENGTH OF BERTH_CRANES)
      END
    END
```

```
@  Crane-allocation for a multiple berth  @
IF BERTH_SINGLE = FALSE
  FOR EACH CRANE IN BERTH_CRANES WITH (CRANE_X_MIN ≤ SHIP_X_BOW) ^
    (CRANE_X_MAX ≥ SHIP_X_STERN)
    SHIP_CRANE_SUPPLY ← SHIP_CRANE_SUPPLY + 1
  END
  @  Allocate cranes if supply ≤ demand  @
  IF SHIP_CRANE_SUPPLY ≤ CLASS_CRANE_DEMAND
    FOR EACH CRANE IN BERTH_CRANES WITH (CRANE_X_MIN ≤ SHIP_X_BOW) ^
      (CRANE_X_MAX ≥ SHIP_X_STERN)
      IF CRANE_AVAILABILITY = TRUE
        SHIP_UNLOAD_RATE ← SHIP_UNLOAD_RATE + CRANE_UNLOADCAP x
                           SHIP_FORTYFEETFAC_IMP
        SHIP_LOAD_RATE   ← SHIP_LOAD_RATE + CRANE_LOADCAP x
                           SHIP_FORTYFEETFAC_EXP
        CRANE_MYSHIP     ← THIS SHIP
        CRANE_X_MIN      ← SHIP_X_BOW
        CRANE_X_MAX      ← SHIP_X_STERN
        ACTIVATE THIS CRANE FROM CRANESTART IN CRANEMOD
      END
      IF CRANE_AVAILABILITY = FALSE
        CRANE_PREV_SHIP ← CRANE_MYSHIP
        CRANE_MYSHIP    ← THIS SHIP
      END
    END
    RIGHTCRANE ← FIRST CRANE IN BERTH_CRANES WITH CRANE_MYSHIP IS THIS
    SHIP
  END
  @  Possibility to shift cranes if supply < demand  @
  IF SHIP_CRANE_SUPPLY < CLASS_CRANE_DEMAND
    TM_SHIP_A ← NONE
    TM_SHIP_B ← NONE
    TM_SHIP_C ← NONE
    TM_LOAD_A ← 1E10
    TM_LOAD_B ← 1E10
    IF PRED OF THIS SHIP IN BERTH_SHIPS IS NOT NONE
      TM_SHIP_A ← PRED OF THIS SHIP IN BERTH_SHIPS
      TM_LOAD_A ← SHIP_IMPORTTOTAL OF TM_SHIP_A + SHIP_EXPORTTOTAL OF
      TM_SHIP_A
      IF ((CLASS_CRANE_DEMAND OF RIGHTCLASS OF TM_SHIP_A -
        SHIP_CRANE_SUPPLY OF TM_SHIP_A) > CLASS_CRANE_DEMAND -
        SHIP_CRANE_SUPPLY) v (SHIP_CRANE_SUPPLY OF TM_SHIP_A ≤ 1) v
        (SHIP_LOAD OF TM_SHIP_A ≥ 0.75 x TM_LOAD_A) v
        ((CLASS_GEAR_LOADCAP + CLASS_GEAR_UNLOADCAP) = 0)
```

```
72            TM_SHIP_A ← NONE
73            TM_LOAD_A ← 1E10
74          END
75        END
76        IF SUCC OF THIS SHIP IN BERTH_SHIPS IS NOT NONE
77          TM_SHIP_B ← SUCC OF THIS SHIP IN BERTH_SHIPS
78          TM_LOAD_B ← SHIP_IMPORTTOTAL OF TM_SHIP_B + SHIP_EXPORTTOTAL OF
            TM_SHIP_B
79          IF ((CLASS_CRANE_DEMAND OF RIGHTCLASS OF TM_SHIP_B -
              SHIP_CRANE_SUPPLY OF TM_SHIP_B) > CLASS_CRANE_DEMAND-
              SHIP_CRANE_SUPPLY) v (SHIP_CRANE_SUPPLY OF TM_SHIP_B ≤ 1) v
              (SHIP_LOAD OF TM_SHIP_B ≥ 0.75 x TM_LOAD_B) v
              ((CLASS_GEAR_LOADCAP + CLASS_GEAR_UNLOADCAP) = 0)
80            TM_SHIP_B ← NONE
81            TM_LOAD_B ← 1E10
82          END
83        END
84        TM_LOAD_C ← MIN(TM_LOAD_A,TM_LOAD_B)
85        IF TM_LOAD_C < SHIP_IMPORTTOTAL + SHIP_EXPORTTOTAL
86          BERTH_CRANE_SHIFTS ← BERTH_CRANE_SHIFTS + 1
87          TM_SHIP_C ← FIRST SHIP IN BERTH_SHIPS WITH (SHIP_IMPORTTOTAL +
            SHIP_EXPORTTOTAL) = TM_LOAD_C
88          IF TM_SHIP_C IS NOT NONE
89            THIS CRANE ← LAST CRANE IN BERTH_CRANES WITH CRANE_MYSHIP IS
              TM_SHIP_C IF TM_SHIP_C IS TM_SHIP_A
90            THIS CRANE ← FIRST CRANE IN BERTH_CRANES WITH CRANE_MYSHIP IS
              TM_SHIP_C IF TM_SHIP_C IS TM_SHIP_B
91            SHIP_RATE OF TM_SHIP_C ← SHIP_RATE OF TM_SHIP_C -
              CRANE_UNLOADCAP x SHIP_FORTYFEETFAC_IMP OF TM_SHIP_C
              IF SHIP_UNLOADING OF TM_SHIP_C = TRUE
92            SHIP_LOAD_RATE OF TM_SHIP_C ← SHIP_LOAD_RATE OF TM_SHIP_C -
              CRANE_LOADCAP x SHIP_FORTYFEETFAC_EXP OF TM_SHIP_C IF
              SHIP_UNLOADING OF TM_SHIP_C = TRUE
93            SHIP_RATE OF TM_SHIP_C ← SHIP_RATE OF TM_SHIP_C- CRANE_LOADCAP
              x SHIP_FORTYFEETFAC_EXP IF SHIP_LOADING OF TM_SHIP_C = TRUE
94            SHIP_CRANE_SUPPLY OF TM_SHIP_C ← SHIP_CRANE_SUPPLY OF
              TM_SHIP_C - 1
95            SHIP_UNLOAD_RATE ← SHIP_UNLOAD_RATE + CRANE_UNLOADCAP x
              SHIP_FORTYFEETFAC_IMP
96            SHIP_LOAD_RATE   ← SHIP_LOAD_RATE + CRANE_LOADCAP x
              SHIP_FORTYFEETFAC_EXP
97            SHIP_CRANE_SUPPLY ← SHIP_CRANE_SUPPLY + 1
98            CRANE_X_MIN  ← SHIP_X_BOW
99            CRANE_X_MAX  ← SHIP_X_STERN
```

```
          CRANE_MYSHIP ← THIS SHIP
        END
      END
      RIGHTCRANE ← FIRST CRANE IN BERTH_CRANES WITH CRANE_MYSHIP IS THIS
      SHIP
    END
    @  Allocate cranes if supply > demand  @
    IF SHIP_CRANE_SUPPLY > CLASS_CRANE_DEMAND
      TM_SHIP_A ← PRED OF THIS SHIP IN BERTH_SHIPS
      TM_SHIP_B ← SUCC OF THIS SHIP IN BERTH_SHIPS
      IF (TM_SHIP_A IS NONE) ^ (SHIP_X_BOW ≤ 100)
        RIGHTCRANE ← FIRST OF BERTH_CRANES
      END
      IF (TM_SHIP_A IS NONE) ^ (SHIP_X_BOW > 100)
        RIGHTCRANE ← FIRST CRANE IN BERTH_CRANES WITH CRANE_NUMBER =
        CEIL(LENGTH OF BERTH_CRANES x SHIP_X_BOW ÷ BERTH_LENGTH)
        IF TM_SHIP_B IS NOT NONE
          IF (RIGHTCRANE OF TM_SHIP_B IS NOT NONE) ^ (100 >SHIP_X_BOW OF
            TM_SHIP_B - SHIP_X_STERN)
            RIGHTCRANE ← FIRST CRANE IN BERTH_CRANES WITH CRANE_NUMBER =
            CRANE_NUMBER OF RIGHTCRANE OF TM_SHIP_B - CLASS_CRANE_DEMAND
          END
        END
        IF (TM_SHIP_B IS NONE) ^ ((BERTH_LENGTH - SHIP_X_STERN) < 100)
          RIGHTCRANE ← FIRST CRANE IN BERTH_CRANES WITH CRANE_NUMBER =
          LENGTH OF BERTH_CRANES + 1 - CLASS_CRANE_DEMAND
        END
      END
      IF TM_SHIP_A IS NOT NONE
        IF RIGHTCRANE OF TM_SHIP_A IS NONE
          RIGHTCRANE ← FIRST CRANE IN BERTH_CRANES WITH CRANE_NUMBER =
          CEIL(LENGTH OF BERTH_CRANES x SHIP_X_BOW ÷ BERTH_LENGTH)
          IF TM_SHIP_B IS NOT NONE
            IF (RIGHTCRANE OF TM_SHIP_B IS NOT NONE) ^ (100 > SHIP_X_BOW
              OF TM_SHIP_B - SHIP_X_STERN)
              RIGHTCRANE ← FIRST CRANE IN BERTH_CRANES WITH CRANE_NUMBER
              = CRANE_NUMBER OF RIGHTCRANE OF TM_SHIP_B -
              CLASS_CRANE_DEMAND
            END
          END
          IF (TM_SHIP_B IS NONE) ^ ((BERTH_LENGTH - SHIP_X_STERN) < 100)
            RIGHTCRANE ← FIRST CRANE IN BERTH_CRANES WITH CRANE_NUMBER =
            LENGTH OF BERTH_CRANES + 1 - CLASS_CRANE_DEMAND
          END
      END
    END
    IF TM_SHIP_A IS NOT NONE
      IF RIGHTCRANE OF TM_SHIP_A IS NOT NONE
        RIGHTCRANE ← FIRST CRANE IN BERTH_CRANES WITH CRANE_NUMBER =
        CRANE_NUMBER OF RIGHTCRANE OF TM_SHIP_A + SHIP_CRANE_SUPPLY OF
        TM_SHIP_A
      END
    END
    GOTO SKIP IF RIGHTCRANE IS NONE
    THIS CRANE ← RIGHTCRANE
    FOR I ← 1 TO CLASS_CRANE_DEMAND
      IF CRANE_AVAILABILITY = TRUE
        SHIP_UNLOAD_RATE ← SHIP_UNLOAD_RATE + CRANE_UNLOADCAP x
        SHIP_FORTYFEETFAC_IMP
        SHIP_LOAD_RATE   ← SHIP_LOAD_RATE + CRANE_LOADCAP x
        SHIP_FORTYFEETFAC_EXP
        CRANE_MYSHIP     ← THIS SHIP
        CRANE_X_MIN      ← SHIP_X_BOW
        CRANE_X_MAX      ← SHIP_X_STERN
        ACTIVATE THIS CRANE FROM CRANESTART IN CRANEMOD
      END
      IF CRANE_AVAILABILITY = FALSE
        CRANE_PREV_SHIP ← CRANE_MYSHIP
        CRANE_MYSHIP    ← THIS SHIP
      END
      IF SUCC OF THIS CRANE IN BERTH_CRANES IS NOT NONE
        THIS CRANE ← SUCC OF THIS CRANE IN BERTH_CRANES
      END
    END
    FOR EACH CRANE IN BERTH_CRANES WITH (CRANE_NUMBER <CRANE_NUMBER OF
      RIGHTCRANE) ^ (CRANE_X_MAX > SHIP_X_BOW)
      CRANE_X_MAX ← SHIP_X_BOW
    END
    FOR EACH CRANE IN BERTH_CRANES WITH (CRANE_NUMBER ≥CRANE_NUMBER OF
      RIGHTCRANE + CLASS_CRANE_DEMAND) ^ (CRANE_X_MIN < SHIP_X_STERN)
      CRANE_X_MIN ← SHIP_X_STERN
    END
    SHIP_CRANE_SUPPLY ← CLASS_CRANE_DEMAND
    SKIP:
  END
  IF RIGHTCRANE IS NONE
    THIS CRANE ← FIRST CRANE IN BERTH_CRANES
    SHIP_UNLOAD_RATE ← SHIP_FORTYFEETFAC_IMP x CRANE_UNLOADCAP x
```

```
           CLASS_CRANE_DEMAND
172        SHIP_LOAD_RATE ← SHIP_FORTYFEETFAC_EXP x CRANE_LOADCAP x
           CLASS_CRANE_DEMAND
173        SHIP_CRANE_SUPPLY ← CLASS_CRANE_DEMAND
174      END
175    END
176    SHIP_CRANE_ALLOC ← TRUE
177    REACTIVATE THIS SHIP
178  END
179  REPEAT FROM TM_START
180
181 TM_REALLOCATE:
182  @  Crane-realloation (when a ship departs)  @    @  regel 180
183  THIS SHIP  ← LAST OF DEPARTINGSHIPS
184  THIS BERTH ← RIGHTBERTH
185  IF BERTH_SINGLE = FALSE
186    THIS SHIP  ← LAST OF DEPARTINGSHIPS
187    THIS BERTH ← RIGHTBERTH
188    FOR EACH CRANE IN BERTH_CRANES WITH (CRANE_MYSHIP IS NONE) v
         (CRANE_MYSHIP IS LAST OF DEPARTINGSHIPS)
189      THIS SHIP ← FIRST SHIP IN BERTH_SHIPS WITH SHIP_X_BOW ≥ CRANE_X_MAX
190      CRANE_X_MAX ← SHIP_X_BOW IF THIS SHIP IS NOT NONE
191      CRANE_X_MAX ← BERTH_LENGTH IF THIS SHIP IS NONE
192      THIS SHIP ← LAST SHIP IN BERTH_SHIPS WITH SHIP_X_STERN ≤ CRANE_X_MIN
193      CRANE_X_MIN ← SHIP_X_STERN IF THIS SHIP IS NOT NONE
194      CRANE_X_MIN ← 0 IF THIS SHIP IS NONE
195    END
196    THIS SHIP  ← LAST OF DEPARTINGSHIPS
197    THIS BERTH ← RIGHTBERTH
198    THIS TERMINAL ← RIGHTTERM
199    THIS CRANE ← RIGHTCRANE
200    GOTO SKIP2 IF THIS CRANE IS NONE
201    TM_SHIP_A  ← FIRST SHIP IN BERTH_SHIPS WITH SHIP_X_STERN = CRANE_X_MIN
202    IF TM_SHIP_A IS NOT NONE
203      IF (SHIP_CRANE_SUPPLY OF TM_SHIP_A < CLASS_CRANE_DEMAND OF
           RIGHTCLASS OF TM_SHIP_A) ^ (SHIP_LOAD OF TM_SHIP_A < 0.75 x
           SHIP_IMPORTTOTAL OF TM_SHIP_A + SHIP_EXPORTTOTAL OF TM_SHIP_A) ^
           ((SHIP_LOADING OF TM_SHIP_A = TRUE) v (SHIP_UNLOADING OF TM_SHIP_A
            = TRUE))
204        SHIP_RATE OF TM_SHIP_A ← SHIP_RATE OF TM_SHIP_A + CRANE_UNLOADCAP
           x SHIP_FORTYFEETFAC_IMP OF TM_SHIP_A IF SHIP_UNLOADING OF
           TM_SHIP_A = TRUE
205        SHIP_LOAD_RATE OF TM_SHIP_A ← SHIP_LOAD_RATE OF TM_SHIP_A +
           CRANE_LOADCAP x SHIP_FORTYFEETFAC_EXP OF TM_SHIP_A IF
```

```
        SHIP_UNLOADING OF TM_SHIP_A = TRUE
        SHIP_RATE OF TM_SHIP_A ← SHIP_RATE OF TM_SHIP_A + CRANE_LOADCAP
        x SHIP_FORTYFEETFAC_EXP OF TM_SHIP_A IF SHIP_LOADING OF
        TM_SHIP_A = TRUE
        SHIP_CRANE_SUPPLY OF TM_SHIP_A ← SHIP_CRANE_SUPPLY OF TM_SHIP_A+1
        CRANE_MYSHIP      ← TM_SHIP_A
        CRANE_X_MIN       ← SHIP_X_BOW OF TM_SHIP_A
        CRANE_X_MAX       ← SHIP_X_STERN OF TM_SHIP_A
        ACTIVATE THIS CRANE FROM CRANESTART IN CRANEMOD
      END
    END
    THIS SHIP  ← LAST OF DEPARTINGSHIPS
    THIS BERTH ← RIGHTBERTH
    THIS TERMINAL ← RIGHTTERM
    THIS CRANE ← FIRST CRANE IN BERTH_CRANES WITH CRANE_NUMBER =
    CRANE_NUMBER OF RIGHTCRANE + SHIP_CRANE_SUPPLY - 1
    GOTO SKIP2 IF THIS CRANE IS NONE
    TM_SHIP_B  ← FIRST SHIP IN BERTH_SHIPS WITH SHIP_X_BOW = CRANE_X_MAX
    IF TM_SHIP_B IS NOT NONE
      IF (SHIP_CRANE_SUPPLY OF TM_SHIP_B < CLASS_CRANE_DEMAND OF
        RIGHTCLASS OF TM_SHIP_B) ∨ (SHIP_LOAD OF TM_SHIP_B < 0.75 x
        SHIP_IMPORTTOTAL OF TM_SHIP_B + SHIP_EXPORTTOTAL OF TM_SHIP_B) ∧
        ((SHIP_LOADING OF TM_SHIP_B = TRUE) ∨ (SHIP_UNLOADING OF TM_SHIP_B
        = TRUE))
        SHIP_RATE OF TM_SHIP_B ← SHIP_RATE OF TM_SHIP_B + CRANE_UNLOADCAP
        x SHIP_FORTYFEETFAC_IMP OF TM_SHIP_B IF SHIP_UNLOADING OF
        TM_SHIP_B = TRUE
        SHIP_LOAD_RATE OF TM_SHIP_B   ← SHIP_LOAD_RATE OF TM_SHIP_B +
        CRANE_LOADCAP x SHIP_FORTYFEETFAC_EXP OF TM_SHIP_B IF
        SHIP_UNLOADING OF TM_SHIP_B = TRUE
        SHIP_RATE OF TM_SHIP_B ← SHIP_RATE OF TM_SHIP_B + CRANE_LOADCAP
        x SHIP_FORTYFEETFAC_EXP OF TM_SHIP_B IF SHIP_LOADING OF
        TM_SHIP_B = TRUE
        SHIP_CRANE_SUPPLY OF TM_SHIP_B ← SHIP_CRANE_SUPPLY OF TM_SHIP_B+1
        CRANE_MYSHIP      ← TM_SHIP_B
        CRANE_X_MIN       ← SHIP_X_BOW OF TM_SHIP_B
        CRANE_X_MAX       ← SHIP_X_STERN OF TM_SHIP_B
        ACTIVATE THIS CRANE FROM CRANESTART IN CRANEMOD
      END
    END
    SKIP2:
  END
  REPEAT FROM TM_START
```