Traffic & Transport
Anna van Buerenplein 1
2595 DA Den Haag
P.O. Box 96800
2509 JE The Hague
The Netherlands

www.tno.nl

T +31 88 866 00 00

TNO report

TNO 2020 R10565

Swarmport agent-based simulation model description and documentation

Date	21 January 2021
Author(s)	Ruben Fransen, Paul Tilanus, Jins de Jong, Igor Davydenko
No. of copies	2020-STL-RAP-100337557
Number of pages	85 (incl. appendices)
Number of appendices	4
Sponsor	NWO/TKI Dinalog
Project name	SWARMPORT
Project number	060.26906

Summary

Development of port nautical chain simulation model
The SWARMPORT project modeled the infrastructure and the nautical services chain of the Port of Rotterdam as an Agent-based Simulation Model (ABSM). This ABSM can be used in scenario-wise studies on the effect of alternative behavior of stakeholders in the nautical services chain and its impact on a number of Key Performance Indicators (KPIs) of the port. The Port Authority can use the developed modelling capabilities together with scenario studies to assess the effects of changes on port performance further extending their Decision Support System.

Can complexity theory support decision making in the Port of Rotterdam?
The main question at the start of this research project was: Can complexity theory be used to describe and model the port nautical chain processes to support decision making at the Port of Rotterdam? The port nautical chain is a multi-stakeholder system where minor alterations in actor behavior can have a significant impact on the system performance. The general port performance is an important factor for port attractiveness and therefore essential for the Port of Rotterdam in the competing with the other ports in the range Hamburg-Le-Havre of deep sea ports. The developed simulation model shows that with an agent-based approach a first version (a proof of the concept) of a decision support model can be developed. However, further research and development of the model is required to increase realism of the modeling implementation of the nautical chain processes at the port of Rotterdam.

Quantifying impact of changes in the port nautical chain
The developed agent-based model shows that impact of scenarios on the system and actor parameters can be analyzed and studied. The study therefore shows that a complexity approach can be used to increase understanding of the nautical chain performance and importantly that the model can be used to quantify impact of measures at the system and actor level. As every model it is a limited representation of reality and therefore a decent level of understanding of the model and the model scope is advised for setting up of the model and interpretation of scenario results.

Contents

1 Introduction

Seaports provide a range of services that together support the port handling process of ships, including positioning, piloting, mooring, offloading and loading of the cargo, bunkering or fueling. As a short and predictable turnaround time is one of the key factors determining the competitiveness of ports, a well-organized chain of nautical services is essential. The performance of this service chain will depend on the dynamic nature of the demand for services (volume and size of ships), on external circumstances (e.g. weather), the capabilities of individual agents within the chain as well as on the collaboration between them. Performance can be enhanced in different ways, including through process agreements and agreements on performance targets, information exchange, and regulation. The development of strategies to increase performance, as well as acceptance and support for these strategies, can benefit from quantitative models that support the evaluation of proposals for possible strategies. Due to the complexity of the system of services it is not a trivial task to create such models.

1.1 Background

The origin of the SWARMPORT project is in bundling of the knowledge and expertise of the Universities, TNO and industrial partners that resulted in a successful project proposal for the NWO complexity program. The SWARMPORT project has been built on the Early Research Program (ERP) Complexity of TNO, complex systems of the ‘Swarm lab’ at the Maastricht University, knowledge on transport and logistics organization at the TU Delft. These initiatives all study complex systems, i.e. systems where the relation between input and output can be highly non-linear and uncertain: small changes in the input can have large effects on the output.

The port nautical service chain is a complex system where moderate changes in environment or in stakeholder behavior – ship agents, captains, pilots, tugboats, terminals, etc. – can cause a significant change in the nautical service chain performance. And it is hard to describe the system dynamics in an analytical model.

The combination of complexity studies and the port nautical service chain has led to the SWARMPORT project. This report describes the TNO approach to deal with the complexity: an Agent-based Simulation Model (ABSM).

An ABSM is a simulation model where the behavior of a stakeholder is modeled as an ‘agent’: software that describes the behavior of the stakeholder. The idea is that a change in behavior of a certain stakeholder can be implemented in the software agent representing that stakeholder. Running the ABSM subsequently with multiple instances of each type of agent, representing multiple ships, multiple pilots, multiple tugboats, etc., shows the effect of the behavioral change on the performance of the whole system of interacting agents/stakeholders, typically measured by a series of Key Performance Indicators (KPIs).

1.2 Swarmport project

In this project knowledge from various partners on complexity science is mobilized and merged with logistic research to develop a simulation model for the nautical chain of the Port of Rotterdam.

The Swarmport project consortium consists of the following parties:
TU Delft
- Maastricht University
- TNO Sustainable Transport and Logistics
- Port of Rotterdam
- Hutchison Ports ECT Delta
- Intertransis
- Smartport

The aim of this project is threefold:
1. To improve our understanding of the self-organizational properties of the chain of nautical handling processes directed at maritime ships around seaports, from arrival to departure.
2. To develop valid and practicable methods for modelling port operational processes, implementing agent-based modelling from a self-organizational, complex system perspective.
3. To design strategies based on self-organizational properties of port processes to increase the resilience, reliability and flexibility of services of individual actors and of the aggregate service chain.

1.3 Reading guide

This document describes the ABSM:
- The nautical service chain information it is based upon;
- The mapping of stakeholders to software agents;
- The manual for running the ABSM;
- The software documentation on how to do changes:
    - Changes in parameters
    - Changes in software agent behavior
    - Changes in the system

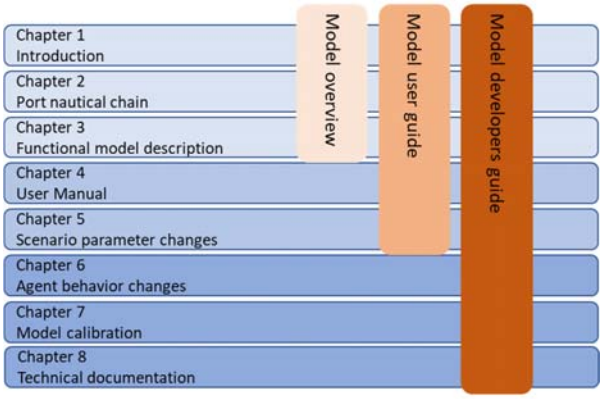

Figure 1: Reading guide overview of chapters and intended reader.

This documentation provides guidance on how to run the ABSM, including what-if scenario analysis. The remainder of this report is structured as follows.

Chapter 2 sets the scene by describing the Port of Rotterdam and the nautical service chain.

Chapter 3 describes the functional model, it provides a functional description of the agents and agent interactions which have not been modelled. For example the mapping of the parties in nautical service chain onto software agents, and the mapping or the communication between parties onto communication between the agents

Chapter 4 is the manual for the model: how to run it, how to interpret the output files and how to use these in a what-if analysis.

The chapters 5, 6 and 8 describe how to change the behaviors in the ABMS at three levels of complexity:

- Chapter 5: Changing parameters (simple);
- Chapter 6: Changing agent behavior (moderately complex, requires Python programming knowledge);
- Chapter 8 : Changing the architecture (complex). This level of changes is assumed to be rare, but might happen when e.g. a new agent is introduced or the ABSM is connected to some other software system.

References and a list of abbreviations can be found in chapter 9 and 10 respectively.

2 General description of port nautical processes

2.1 Port of Rotterdam

Figure 2: Map of the Rotterdam port. Source Port of Rotterdam Harbor master

Sea going vessels make over 29.000 port calls to the Port of Rotterdam per year[1]. This makes it one of the largest deep sea ports in the North-western Europe, also known as the Hamburg-Le-Havre range.

All the vessels visiting the port, port calls and passers-by, have to commit to the local rules and procedures of the port, as defined by the European and Dutch governments, and Port Authority. More information can be found in the [Port Information Guide, 2020]. The processes and services that together ensure safe and eventless port visits are called the port nautical service chain. The actors and stakeholders that make up the nautical service chain are elaborated in the next section.

2.2 Nautical service chain

The description of the nautical service chain in this section is based on the thesis of A. Verduijn [Verduijn, 2018], augmented by interviews with the Swarmport consortium partner Intertransis. The nautical service chain is being described by the various service actors that play a role in servicing the journey of vessels through the Port of Rotterdam. The vessels can be seen as 'jobs' for the nautical service chain. Communication between a vessel and the required service is essential for requesting, scheduling and executing services.

2.2.1 Port authority

The Port Authority is represented by the Harbor Master and has the responsibility to maintain safe and smooth handling of shipping operations in the Port of Rotterdam. In the port nautical service chain the Harbor Master has to provide administrative and operational clearance before ship are allowed to enter or leave the port.

[1] https://www.portofrotterdam.com/en/news-and-press-releases/nautical-annual-figures-2019

2.2.2 Pilotage

All vessels over 75 meters long are obliged to have a registered maritime pilot to sail in the Port of Rotterdam. In general the pilot boards the vessel at open sea to guide an incoming vessel to its berth or the pilot boards at the berth and guides the vessel outwards and the pilot disembarks at open sea. There are, however, many different cases situations which deviate slightly from this general concept. A few examples are: ships that sail from one berth to another berth in the port, passer-by ships that do not call at a berth or some skippers have their own pilot registration.

In general the pilots that guide the vessels work for the pilot organization, which, as the names states, organizes the piloting operations. As the vessels are obliged to have a pilot aboard, so is pilot organization obliged to deliver a pilot if requested by a vessel.

Figure 3: l*eft*: Pilot boarding a deep sea vessel at open water. *right*: Tugboats pushing container vessel to the terminal and linesmen boat handling one of the shore lines.

2.2.3 Tugboat companies

Large deep sea vessels have limited maneuvering capabilities on the narrow waters inside ports. Tugboats are therefore required to smoothly and safely maneuver, berth and unberth these vessels. The number of tugboats required per vessel varies with the environmental conditions such as the weather, and size of the ship.

2.2.4 Linesmen

One of the last step of the incoming port nautical chain is to fix the vessel to its berth place, where one of the first steps of the outgoing port nautical chain is to release the vessel from its berth. The linesmen of the Port of Rotterdam is a single organization that ensures this process runs smoothly, by providing staff and the required equipment for the mooring and unmooring process.

2.2.5 Terminals

Most vessels visiting the Port of Rotterdam have a logistical purpose of unloading and/or loading goods at a cargo terminal. As long as a vessel is docked at a terminal or another berth, there are actually no nautical chain processes required. The terminals ensure that there is a berth place for incoming vessels and they have to finish their operations before the vessels is ready to depart for the outgoing nautical service chain. Therefore the role the terminals play is large in the logistical service chain and limited in the nautical service chain.

3 Functional model description

3.1 Why agent-based model

As mentioned in section 1.1, the system of processes that make up the port nautical service chain can be seen as a complex system. From the complex system point of view, a minor change in environment or in actor behavior can cause a significant impact on the nautical service chain operations. In a complex system, a small initial change propagate through the time affecting more and more states of the processes. Likewise, the system dynamics of the nautical chain process, which are influenced by initial values and states of the variables, are hard to define on global level. By defining dynamics and behavior on actor level and simulating interaction of these actors in an agent-based model gives the opportunity to analyze effect of parameters or actor behavior on the global system. In other words it allows scaling up micro states of the system parts to more aggregate parameters, where the unpredictability of the states of individual actors is not important, and where the stable aggregate performance of the system is studied. This approach makes it possible to use a model to study the impact of changes in environment and operational logic on the total performance of the port, such as the average turn around time, unproductive waiting and other relevant performance indicators.

3.2 Implementation of agent-based model – process

Within the Swarmport project research on the port nautical service of the Port of Rotterdam has been done. The start of this research was on describing and analyzing the individual agents and processes in the port nautical service chain.

First process descriptions were derived by TNO with input from project partners Intertransis and the Port of Rotterdam and implemented in a first agent-based model in Netlogo [Davydenko et al., 2019]. More details of the processes and agent behavior were collected by on-site experience and interviewing stakeholders by [Verduijn, 2018]. Analysis of port call data of the Port of Rotterdam and ship location data (AIS) by [Kaljouw, 2019] provided quantitative insight in duration of subprocesses of the port nautical chain. Numerical results from this study and data analysis are used to calibrate the developed simulation model to match real world performance.

To scope the problem decisions had to be made about which processes to add in detail to the model and which to leave out of scope. Together with the Port of Rotterdam choices have been made to scope the problem.

This resulted in that the following actors are implemented as software agents in the ABSM:

- Vessels –representing ship and their ship agents
- Pilots
- Pilot organization
- Tugboats
- Tugboat organizations
- Port Authority

From section 3.4 onward “agent” is used both for the stakeholder and the software agent representing that stakeholder.

This implies that the following actors are not implemented in the ABSM. They have been left out because their operations are assumed to have less influence on the performance of the nautical service chain in the Port of Rotterdam, compared to the stakeholders listed above:

- Linesmen
- Terminals
- Customs
- Bunker vessels

3.3 Scope

This paragraph contains choices made to define the scope of the simulation model. Choices are made in discussion with the Port of Rotterdam and with goal of making a decision support system that simulates the port nautical chain and where the ABMS is the core modelling engine. On the practical level, the scope choices can also be influenced by data availability and limitations: these relate to general data availability and to limitations imposed by data ownership and data protection regulations, which are not expected to change substantially in the midterm perspective

Simulating port nautical chain

- Simulated agents are ships, pilots, pilot organization, tugboats, tugboat organization and the port authority. According to the Port of Rotterdam the linesmen are very rarely play a role that impacts the port nautical chain performance-wise, in general they are always available and always on time. Therefor the choice has been made to make a first simulation model with pilots and tugboats as the only service agents.
  The terminals are not simulated agents. Some terminal parameters are added to the ship simulation agent.
- Only ships that require a pilot and at least one tugboats, and a berth are simulated in the model. These can be ships visiting single or multiple berths (shifters).
- To first model the nautical chain for the majority of vessels it is chosen to simulate vessels which visit the port for loading and unloading purposes and leave the port after spending stochastically determined time slots at berths.

Data availability

- Port call data of 2016 and 2017 was available as input for the simulation. The simulation therefor only contains berths where ships from this data with pilots and tugboats arrived in 2016 and 2017 in Maasvlakte, Europoort, Waalhaven and Botlek.
- The simulation contains basic assignment and scheduling algorithms for the service agents, because no info on real world scheduling and assignment logic was available in the project.

3.4 Nautical chain main processes

This paragraph gives an overview of the three different main nautical chain processes that are implemented in the simulation model.

The three processes are:

1. Ship arrival
2. Ship departure
3. Ship shift

Note that in the real port there are passers-by ships that require nautical chain services, but do not berth in the Port of Rotterdam. These passing ships have not been integrated in this version of the model.

For each of the 3 processes a summary of the process flow has been illustrated in the following subsections, with the main elements and the key agents for the process elements. Note that the mooring service is a main element of the nautical service chain and therefore are added to these main process flows. The mooring service is not modelled explicitly as the service level of the linesmen, so-called “Roeiers”, is in general very good and therefore very rarely disturbing the nautical chain process. Therefore the choice has been made to first get a good grip on the nautical chain process including pilot and tug boat services.

3.4.1 Arrival process

Figure 4 illustrates an abstraction of the arrival process and shows which key agents are involved at which stage of the incoming nautical chain. The mooring service has been lowlighted because it is not part of the simulation model, but it is an essential service in the chain. The scope of the model describes why this service is left out.

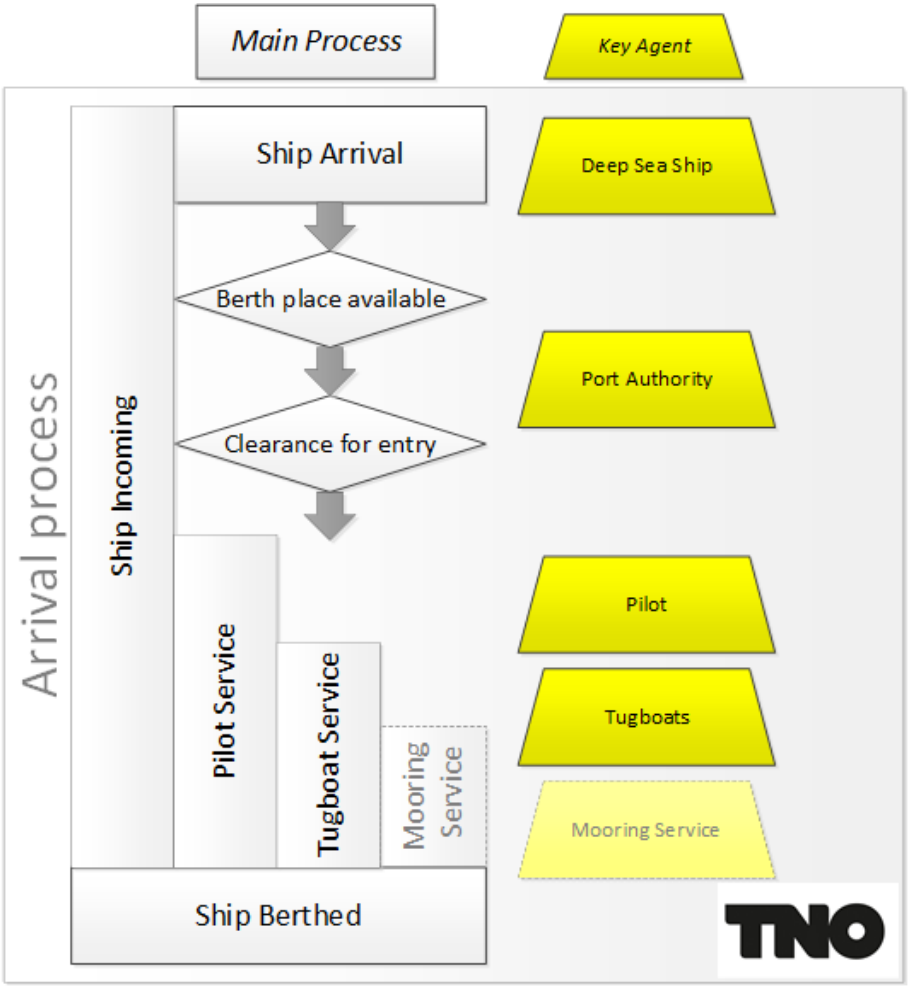

Figure 4: Functional illustration of arrival process and key agents per process step.

3.4.2 *Departure process*

Similar to the picture of the arrival process is an abstraction of the departure process shown in Figure 5. In the departure process the vessel and all the service agents have to be ready simultaneously to start the departure process. This means if one of the actors is delayed that it imposes waiting time for the other actors. This stresses out the importance of availability and on time allocation of the services and processes.

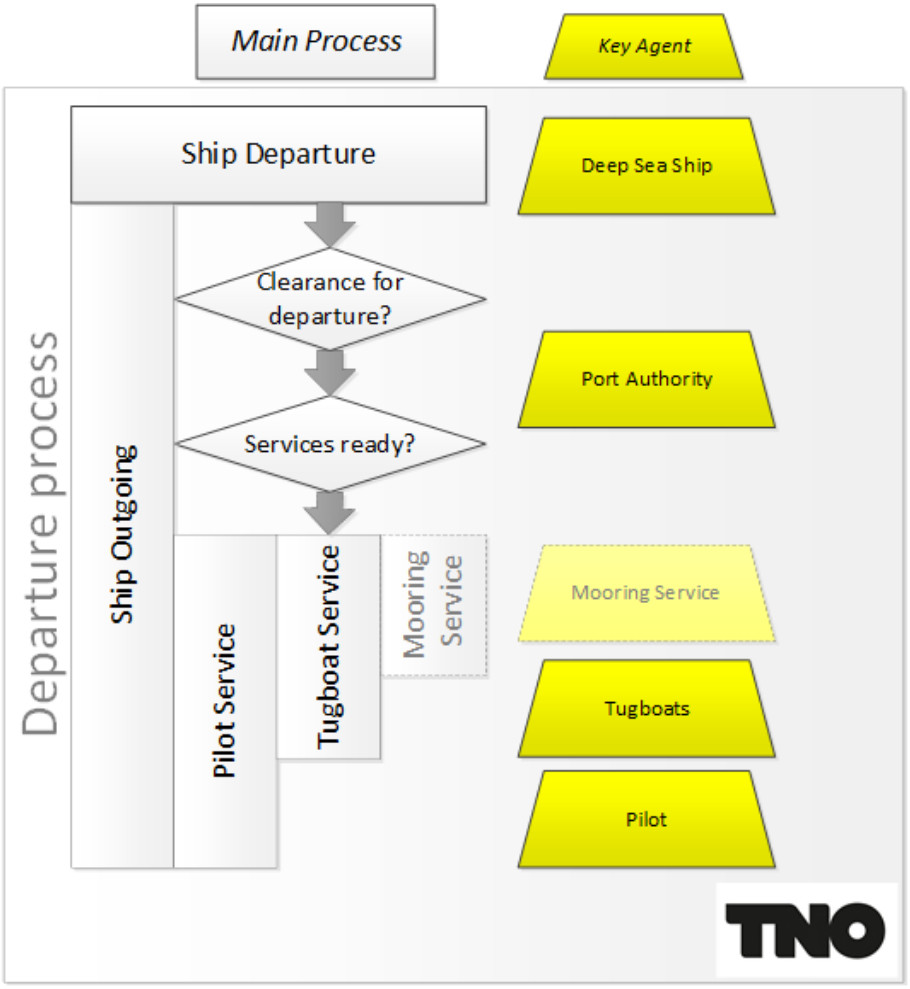

Figure 5: Functional illustration of departure process and key agents per process step.

3.4.3 Shifting process

Figure 6 illustrates the shifting process similar to the departure and arrival process before. This process has a lot in common with the departure process, the main difference is that when this process ends all services also finish simultaneously.

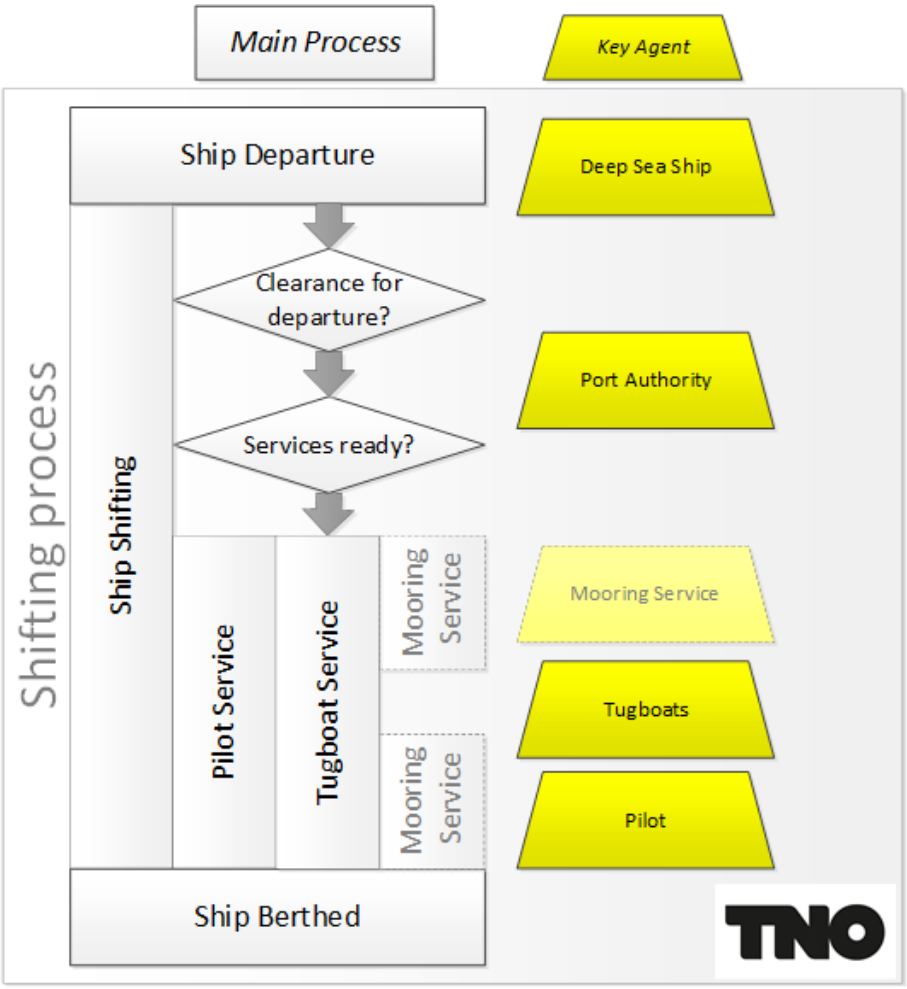

Figure 6: Functional illustration of shifting process and key agents per process step.

3.5 Behavior per agent

The implemented behavior of the different agents is described in this paragraph. Physical agents such as the ships, pilots and tugboats are implemented as “state-machine” agents. These follow certain behavior in each specific state. For these agents the state flow chart is given. For the other agents, the port authority, tugboat company and pilot company the functionality is only described. As they always follow the same operational logic. The functionality of the different communication structures between the agents is described in the next paragraph.

In the agent descriptions below the different states are categorized in three different types regarding the position of the agent:

- Static
- Movement
- Move-along

Note this categorization only holds for agents that have a physical appearance and can therefore move, these are ships, pilots and tugboats. This categorization has been made to structure the simulation process. Which is in detail described in paragraph 8.5 on the simulation core.

3.5.1 Ship

Ships are the key agents of the simulation; each ship is in the simulation represented by one ship agent instance. Ships arrive at the Port of Rotterdam – enter the simulation in state Prebirth – visit one or more terminal berths before they depart to the open sea again – i.e. leave the simulation in state Out of port. To reach and depart from the berth places the services of pilots and tugboats, and clearance from the port authority is required. So the ship agent requires actions from these other agents and therefore the ship agent is a key agent in activating behavior of other agents in the simulation. The state flow for a ship agent is depicted in Figure 7.

3.5.1.1 Prebirth

Initiation of a ship agent in the simulation. In this state the terminal destinations, berth places and time at each berth place are randomly generated based upon the known distributions, for more information on these distribution see paragraph 8.3. In this state the ship requests clearance at the port authority. Clearance can be given by the port authority if the destined berth is available.

3.5.1.2 Call ship agent

In the simulation the ship agent (software) represents both the physical ship and the shipping agent. This step is executed if clearance has been given by the port authority. Execution beholds the ship requesting pilot and tugboat services for his upcoming nautical chain. This is done for each of the different type of nautical chains: incoming, departure or shifting. Depending on the type of the current nautical chain the ship agent is in, the next state for the agent is determined: Sailing to Maascenter buoy, Prepare for shifting or Prepare for outgoing.

3.5.1.3 Sailing to Maascenter buoy (movement)

If at Maascenter buoy, i.e. at the end of the movement duration, and no pilot has been assigned or no clearance has been given by the port authority, then go to the anchor area (state: Incoming anchorage). If pilot is assigned and heading for the ship and clearance granted then proceed to state Incoming pilot.

3.5.1.4 Incoming anchorage (movement)

Sail to anchor area. If reached go to state At anchorage.

3.5.1.5 At anchorage

Wait at anchorage until clearance has been granted and pilot has been assigned.

3.5.1.6 Incoming pilot

Sail to the meeting location of vessel and pilot. If the pilot does has not boarded when the ship gets close to the port then go to state waiting pilot, else go to state Incoming tugboats.

3.5.1.7 Waiting pilot

Wait for pilot to board the vessel, if so go to state Incoming tugboats.

3.5.1.8 Incoming tugboats (movement)
Sail to the tugboat meeting point at the port entrance if destined berth is in Maasvlakte or Calandkanaal or to the meeting point at the city centre for city centre berths. If tugboats are ready go to state Incoming tugboats else go to state Waiting tugboats.

3.5.1.9 Waiting tugboats
Wait until assigned tugboats are at the meeting point. If so go to state Incoming tugboats.

3.5.1.10 Incoming (movement)
Sail to the berth destination then go to state ship berthed procedure.

3.5.1.11 Ship berthed procedure
Signal to tugboats and pilots that their assignment is complete. Go to state At terminal.

3.5.1.12 At terminal
Stay at the terminal for as long as has been determined in Prebirth state, where a stochastic (un)loading time has been determined. A few hours before (un)loading processes have been finished request clearance for next nautical chain: departure or shift. If clearance has been granted then call ship agent, see 3.5.1.2, to request services. If berthing time has been fulfilled go to next state depending on next destination: Prepare for shifting or Prepare for outgoing.

3.5.1.13 Prepare for shifting
Wait for pilot, tugboats, if all are in place then go to state Shifting.

3.5.1.14 Prepare for outgoing
Wait for pilot, tugboats, if all are in place then go to state Outgoing.

3.5.1.15 Shifting (movement)
Sail to the next berth and go the Ship berthed procedure.

3.5.1.16 Outgoing (movement)
Sail out of the port and go to state Out of port

3.5.1.17 Out of port
Ship has left the port, log parameters for simulation performance calculation and clear all simulation resources related to the ship agent.

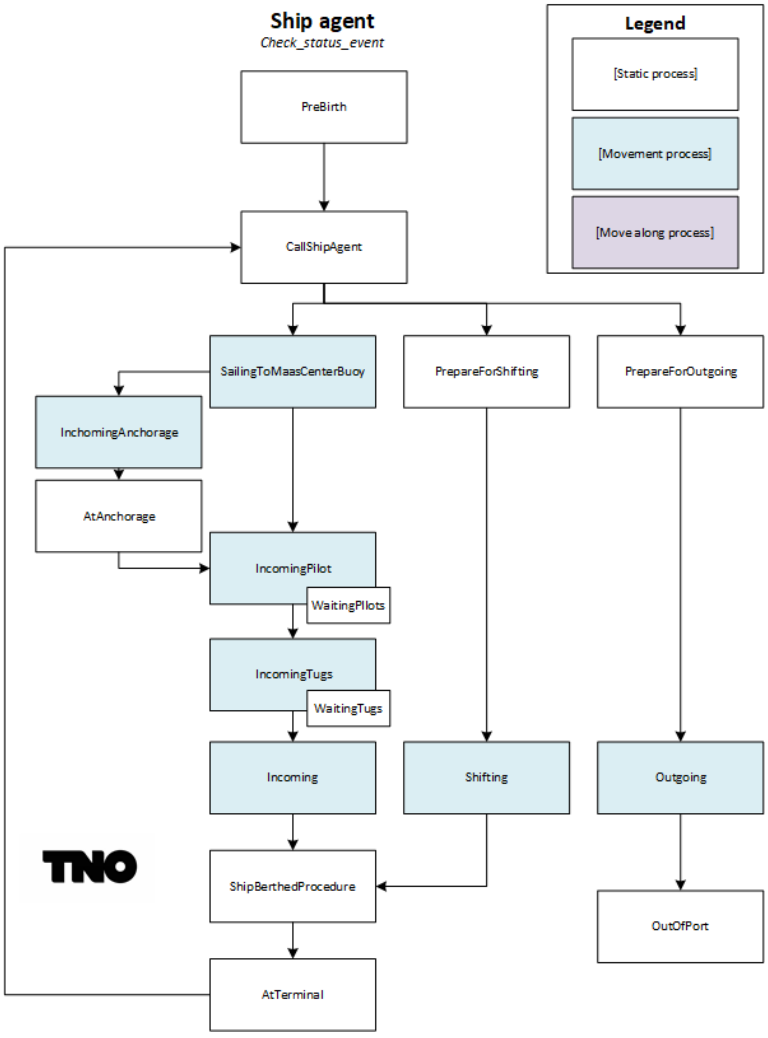

Figure 7: Status flow of ship agent in simulated port nautical chain.

3.5.2 *Port Authority*

The port authority agent represents or can represent a number of real world stakeholders in the simulation model. The agent has the functionality to give clearance for ships to enter into the port, shift within the port, or depart from the port. The only requirement currently in the simulation for ships to get clearance for entrance or shift is that the destination berth place is not occupied. Therefor there is no clearance condition for departure and for departure request clearance is always granted directly.

The implemented functionality of this agent can be described in the following two steps:

1. Listen to clearance-request messages
2. For each request, if the destination berth is empty or has become empty, message to the ships that clearance is granted.

The functionality of the port authority agent can be upgraded if required for future scenario analysis.

3.5.3 Pilot

The pilot agents are one of the two types of service agents in the simulated nautical chain, the other one is the tugboat agent. As can be seen in the state flow of the ship in Figure 7, both service agents are required for all simulated ship movements in the port. Like in practice there are two different pilot directions in the simulation, pilots that pilot incoming vessels and pilots that pilot departing or shifting vessels. The direction of the next pilot voyage depends upon the current one. After an incoming or a shifting voyage a pilot can do an outgoing or shifting voyage. After an outgoing voyage a pilot can do an incoming voyage. In the case of an unbalance in pilots with the required direction for pilot requests, the transfer function to transfer pilots from open sea to the port or vice versa. The state transitions of the pilot agent can be found in Figure 8, and further explained below.

3.5.3.1 Available

The pilot is available at the pilot boat near Maascenter or in the Pistoolhaven and ready for assignment.

3.5.3.2 Assigned

Pilot agent has been assigned and details in the assignment message determine the next state.

3.5.3.3 To Maascenter (movement)

Pilot agent sails from the pilot boat to Maascenter and then to status To ship incoming

3.5.3.4 To ship incoming (movement)

Pilot agent sails from Maascenter to the assigned ship to board the ship and goes to status Incoming.

3.5.3.5 To ship anchorage (movement)

Pilot agent sails from the pilot boat to the assigned ship to board the ship and goes to status Incoming.

3.5.3.6 Incoming (move-along)

In this state the pilot agent location is identical to the location of the incoming ship. The agent remains in this state until the assignment is complete, then the pilot agent state proceeds to status Returning.

3.5.3.7 Returning (movement)

Return to the pilot boat if the last assignment was an Outgoing ship or Pistoolhaven if the last completed assignment was an Incoming or Shifting ship.

3.5.3.8 To terminal (movement)

The pilot agent goes from Pistoolhaven to the assigned ship to board the ship and goes to status Outgoing or Shifting.

3.5.3.9 Shifting (move-along)

In this state the pilot agent location is identical to the location of the shifting ship. The agent remains in this state until the assignment is complete, then the pilot agent state proceeds to status Returning.

3.5.3.10 *Outgoing (move-along)*

In this state the pilot agent location is identical to the location of the outgoing ship. The agent remains in this state until the assignment is complete, then the pilot agent state proceeds to status Returning.

3.5.3.11 *Transfer (movement)*

The general route for pilot services is doing an Outgoing voyage and subsequently and Incoming voyage. Therefore the directions Incoming and Outgoing are taken into account when pilots are being assigned. Shifting voyages can be done after and/or before Outgoing voyages, as the pilots stay in the port. It is possible that there is an inbalance in incoming and outgoing ships and that pilots are needed for incoming vessels while they are all inside the port. In that case transfer assignments are needed to make in the port 'Outgoing' pilots being transferred to open sea to do an Incoming voyage. Vice versa this is also possible.

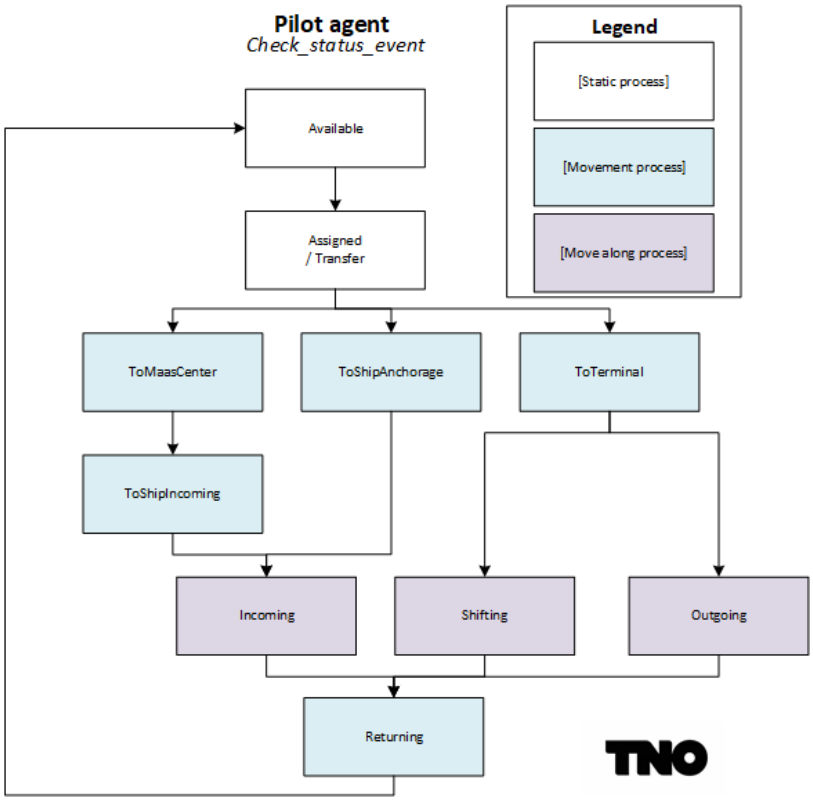

Figure 8: Status flow of pilot agent in simulated port nautical chain.

3.5.4 *Pilot company*

The pilot company agent in the simulation model is responsible for assignment of pilot agents to pilot requests from ships. This pilot scheduler listens to pilot service requests and assigns available pilots to these requests.

There are two different type of pilot assignments:

1. Incoming pilots – piloting incoming ships from Maascenter to a berth place
2. Outgoing pilots – piloting ships from their berth place either to:
    a. Maascenter for an outgoing vessel;
    b. Next berth for a shifting vessel.

The currently implemented assignment method differs from the real-world assignment method. Due to the sensitivity of this research contact with the service providers for this research has been limited, therefore the insight in real world assignment methodology lacks in this research. For practicality the choice has been made to assign pilots on a first-come first-serve (FCFS) basis. The simulation model could be extended with a more intelligent heuristic on scheduling pilots to service assignments, taking into account more details and optimizing the scheduling towards a certain objective. The problem can be approached as a job-scheduling problem with a challenging geographical aspect, it is quite similar to taxi allocation problems. The current modelling setup based on the FCFS assignment of agents probably requires a larger number of pilot agents for the same level of service than in reality. This issue is alleviated in calibration process.

3.5.5 Tugboat

The tugboat agent is the second of the two types of service agents in the simulation. Different from the pilot agent the tugboat assignments are not categorized by direction. Tugboats just get tug assignments from one place in the port to another, from a start point tugging to a berth, or vice versa or for shifters from berth to berth. Note that the location of start of tugging depends on the location of the berth in the port of Rotterdam. When not on duty, when the tugboat state is available, they rest at so called resting locations, see the description of the tugboat scheduling by the tugboat company and Figure 10: The 32 areas in the Port of Rotterdam which occupy both the set of potential resting locations and the set of demand and return points. The red crosses indicate the current locations of the resting facilities. Source: [Kaljouw, 2019]. Figure 10 for these resting locations.

3.5.5.1 Available

A tugboat agent in state available represents a tugboat at one of the tugboat resting locations and ready for assignment.

3.5.5.2 Assigned

Tugboat agent has been assigned and goes to status To start point tugging

3.5.5.3 To start point tugging (movement)

Tugboat agents sails to start point tugging.

3.5.6 To ship (movement)

If assignment ship is getting close enough, sail towards the ship. When arrived at the ship connect the tugboat and go to status Tugging.

3.5.6.1 Tugging (move-along)

In this state the tugboat agent location is identical to the location of the assigned ship. The tugboat agent remains in this state until the assignment is complete, then the tugboat agent state proceeds to status Returning.

3.5.6.2 Returning (movement)

Tugboat now becomes available for new assignment from the tug scheduler of the tug company. The tugboat sails to the nearest resting location and at arrival goes to status Available.

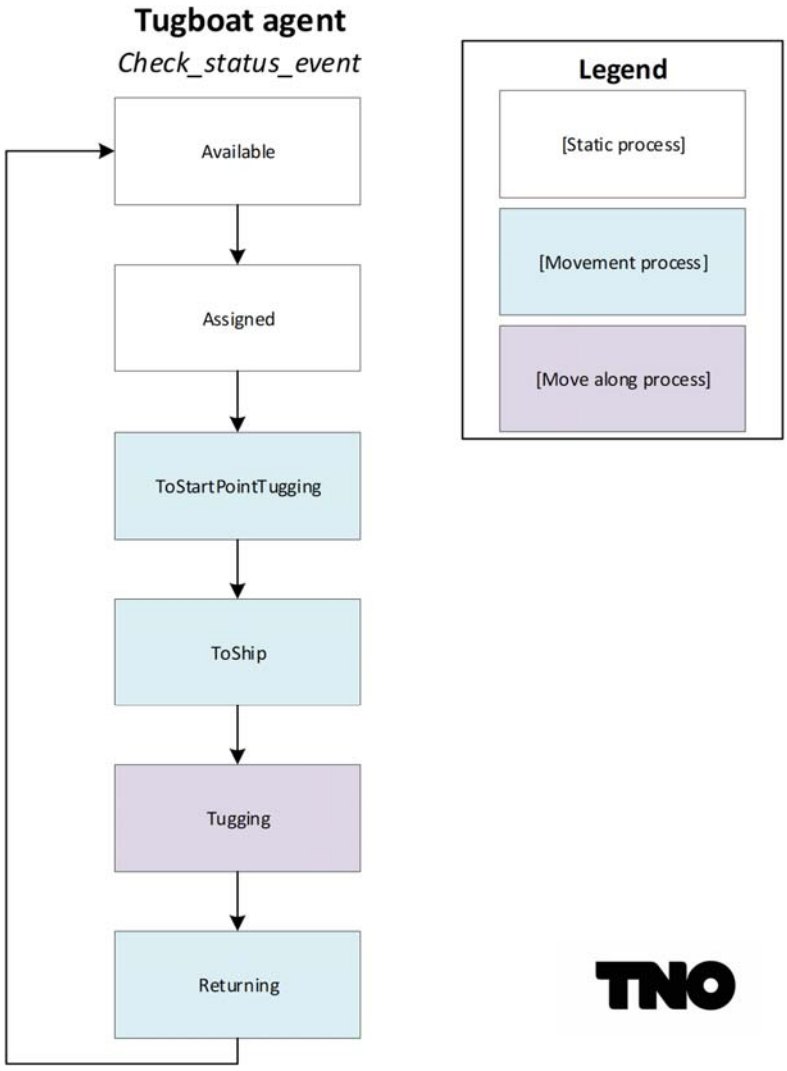

Figure 9: Status flow of tugboat agent in simulated port nautical chain.

3.5.7 Tug company

The tug company agent in the simulation model is responsible for assignment of tugboat agents to tugboat requests from ship agents. This tug scheduler listens to tug service requests and assigns available tugboats to these requests.

Different from the pilot scheduling the tugboat services are not disaggregated on ‘incoming’ or ‘outgoing’. Instead they are categorized by their resting location. There are 5 different resting locations dispersed around the port. Where the tug scheduler at first tries to assign tugboats to a tugboat request from the nearest resting location. If none are available, tugboats from other resting locations can be assigned.

Figure 10: The 32 areas in the Port of Rotterdam which occupy both the set of potential resting locations and the set of demand and return points. The red crosses indicate the current locations of the resting facilities. Source: [Kaljouw, 2019].

The currently implemented assignment method differs from the real world assignment method. Given the sensitivity of this research the contact with the service providers for this research has been limited, therefor the insight in real world assignment methodology lacks in this research. For practicality the choice has been made to assign tugboats on a first-come first-serve (FCFS) basis. The simulation model could be extended with a more intelligent heuristic on scheduling tugboats to service assignments, taking into account more details and optimizing the scheduling towards a certain objective. Similarly to the case of pilot company, the assignment rules probably require more modelled tugboat agents than in reality. This issue is deal with in calibration process, estimating the number of modelled tugboats necessary to replicate real world based year service level of the tugging.

Note that the current simulation model implements only one (1) tug company which assigns tugboats to tugboat requests. In reality there are several tugboat companies handling their own contracted vessels with their own tugboats and competing for non-contracted ships.

The reason to implement only one tugboat company is a lack of information, same as for the assignment methods of the service agents not many details are known. Without the details on which vessel-berth combinations are tugged by which tugboat company and the availability of tugboat for the various tugboat companies adding this feature to the simulation model would not increase the analytical value of the model results.

One of the consequences of implementing only one tugboat company is that vessels always get assigned a tugboat that is closes to their request location. Instead of a company specific available tugboat which could be further away. This is a model imperfection which might cause that the simulated ship waiting time for a tugboat could be slightly less in the simulation than it would be in practice. To compensate for this imperfection and other imperfections coming from model choices and assumptions calibration of parameters is done. More details on this calibration can be found in chapter 7.

3.5.8 *Parallel view of agent state flow*

In the agent-based nautical chain model state flows for the various agents are defined.

State flow changes depend on logic per state which can be influenced by:

- environmental conditions (distance to object),
  - e.g. distance to an object
- organizational conditions,
  - e.g. for a ship: clearance from Port Authority
  - e.g. for a pilot: assignment from scheduler
- other agent conditions.
  - e.g. status of another agent

This means that some of the state flow changes depend on state conditions of other agents. To give a clear overview of which agent states are interdependent and how these interdependencies take place in the nautical chain the state flows for the different chain directions have been visualized, in parallel for the agents ship, pilot and tugboat in an arbitrary time scale. The moment where state changes require state conditions in other agents are highlighted by vertical striped lines, which indicates state changes are synchronized at that point in the nautical chain.

Note that in the figure below only agents are incorporated which are activated by the simulation clock. The other agents which are activated by agent to agent communication such as the service agent schedulers and Port Authority are excluded, although they play their parts(organizational conditions) in the processes. See paragraph 5.1.1 for more information on the simulation clock.

3.5.8.1 *Incoming*

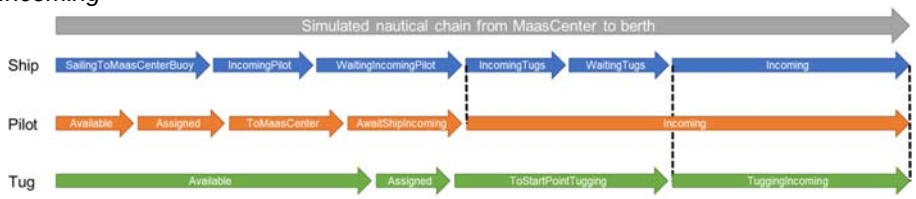

Figure 11: Simulated nautical chain from MaasCenter to berth.

3.5.8.2 *Incoming with anchorage*

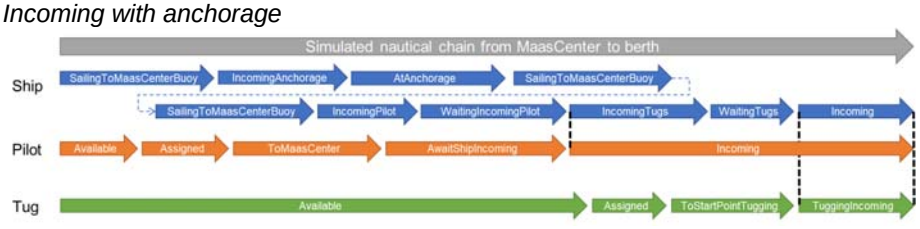

Figure 12: Simulated nautical chain from MaasCenter to berth with anchorage.

3.5.8.3 *Outgoing*

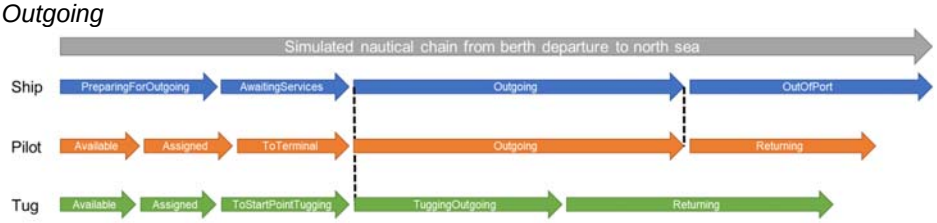

Figure 13: Simulated nautical chain from berth departure to North Sea.

3.5.8.4 *Shifting*

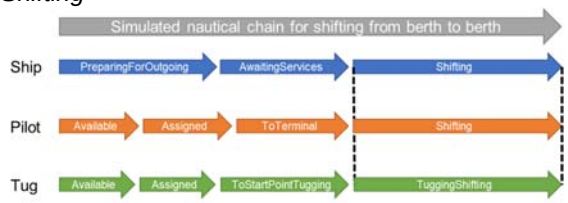

Figure 14: Simulated nautical chain for shifting from berth to berth.

3.6 Communication methods

The simulation model implements the following method of communication between the agents. This paragraph gives a functional description of the implemented communication elements, the technical details can be found in the technical documentation (sections 8.9 and 8.10).

1. HAMIS – HArbor Master Information System
2. VTS – Vessel Traffic management System
3. Direct agent – to – agent

3.6.1 *Harbor master information system: HAMIS*

HAMIS functions as a communication platform to handle all service requests, service assignments, clearance and required information for these organizational features. The list below describes per agent what information is send to or read from the simulated version of HAMIS.

- Ship
  send:
  - Request clearance for entry, shift or departure.
  - Request for pilot service
  - Request for tugboat service including number of required tugs.

  read:
  - Clearance if given
  - Pilot assigned to service ship
  - Tugboats assigned to service ship
- Pilot
  read:
  - Assignment of self to a ship
- Tugboat
  read:
  - Assignment of self to a ship

- Port Authority
  - *send:*
  - o Clearance
  - *read:*
  - o Request for clearance
- Pilot company:
  - *send:*
  - o Assignment of pilot to ship
  - *read:*
  - o Requests for pilot services
  - o Pilot assignments being completed
- Tugboat company:
  - *send:*
  - o Assignment of tugboat to ship
  - *read:*
  - o Requests for tugboat services
  - o Tugboat assignments being completed

3.6.2 Vessel traffic management system: VTS

Communication platform that records all locations of ship agents, pilot agents and tugboat agents and the status information of these agents.

VTS has two main purposes for the simulation:

1. Location of agents - for example service agents (pilots, tugboats) to sail to the location of their assigned ship.
2. Logging simulation data – in particular state transition times – for port performance analysis of the simulated processes.

The VTS shares messages with following contents from the ship, pilot and tugboat agents:

- Agent ID
- Current location and current destination
- Status
- Assignment information
  - o Assignment for service agents
  - o Assigned service agents for vessels

The organization agents do not post information on the VTS, as they do not a have physical representation or location and do not follow a state flow.

3.6.3 Direct agent to agent communication

Main purpose of this direct communication is to connect service agents which are ‘connected’ to a vessel, which are towing tugboats or an onboard pilot. These ‘connected’ service agents are in a move-along state in which they require a location update from the ship agent to update their own location along with it. Next to location updates this communication is also used for communicating the end of a service assignment. When the vessels reaches it berth place or it leaves the port and reaches the ‘end of tugging location and end pilot service location’, then the service assignments are completed and a ‘assignment-complete’ signal is given via this direct communication line. This direct communication is model implementation of real-world direct communication and line of sight.

3.7 Application of model

The simulation model can be used to estimate the impact of scenario changes on the nautical chain performance. What-if analysis can be executed by determining what-if scenarios and simulating the nautical chain processes in those scenarios. Subsequently the performance from the scenario should be compared to the performance in the reference scenario, also known as “baseline”, to determine the impact of the specified scenario.

3.7.1 Scenario assessment

The simulation model quantifies impact of what-if scenarios onto changes in Key Performance Indicators (KPIs). Assessing and analyzing performance changes caused by the scenario is required to give significance to scenario impact. Important performance indicators are the turnaround time of vessels, the quality of nautical services defined by availability and delays in the nautical chain processes. The KPIs are described in more detail in the next chapter, where this chapter continues on the application of the model in possible scenarios and what alterations are required for different types of scenarios.

3.7.1.1 Scope of scenarios

To give an idea of the type of scenario changes that suit the simulation model, examples of different changes are categorized in non-suitable and suitable changes below. It is important to note that the model is a limited representation of reality and therefore limits the scope of possible scenarios. However, to overcome limitations that prevent analysis of essential nautical service chain scenarios it can be possible to further develop and extend the model.

- Model is not (directly) suitable for:
  - Impact of changes in nautical chain architecture
- Model is suitable for:
  - Impact of changes in the system on the nautical chain process,
  - Duration of chain processes, waiting time for services, waiting time for berth availability; average turn around time of vessels. Note that other operations, (un)loading and bunkering are key factors in turn around time which are not explicitly included in this model .
  - Impact of scale increase, larger vessels on port nautical chain processes
  - Impact of changes in port infrastructure

3.7.2 Examples of scenarios and required model adjustments

Scenarios changes can be made on different aspects of the port nautical chain. Some aspects are more easy to vary as they can be edited in simulation parameters on simulation input data, where others such as adjusting behavior of agents require changes in the agent model software. Examples of parameters of behavior to change in scenarios are given in the following subparagraphs.

3.7.2.1 Scenarios without software adjustments

Elements which can be changed in the input parameters or input data is mostly about the environment of the simulation, e.g. how many terminals, how many ships visiting or how many service agents there are.

- Editing simulation parameters
    - Ship arrival rate
    - Number of service agents in the port
        - Pilots
        - Tugboats
    - Average sailing speeds in the port
- Editing input files
    - Terminals in the port
    - Berths per terminal
    - Distribution of visits per terminal / berth

3.7.2.2 Scenarios with software adjustments

Software adjustments can be made on several levels. The following three levels of changes are identified in a growing order of complexity of implementing these scenarios (note theses three levels are discussed in detail in Chapters 5-6).

1. Changing functionality of single agent type with no extra communication required from other agents. For example, by adding or removing a condition for an agent to move on to the next state. A more specific example is adding the condition that a tugboats must be assigned before a pilot will start to move to a ship that is preparing its departure, the pilot can read required assignment information already from the HAMIS message board.
2. Changing functionality of a process step of one agent which requires extra communication from another agent. Which also implies making adjustment to the communication messages. Adding or removing information from communication stream requires adjustments in the sending agent, but also in the receiving agent. An example in the context of the previous example is that a pilot could only start to move to a departing vessel if the tugboats shared an update of their ETA at the vessel. It would require tugboats to share an ETA update and the pilot to read it.
3. Changing something in the agent architecture, e.g. adding a new type of agent, adding a level of detail to agents which influences agent interaction or adding a new stream of communication between agents. For instance, implementing multiple tugboat companies by creating multiple tugboat company agents at the start of the simulation, add a company label to each tugboat, allowing a tugboat company to assign tugboats to jobs if the tugboat has its own label, and adding a tugboat company label to a ship in the Prebirth state representing the contracted tugboat company.

Making adjustments in the simulation architecture has not be mentioned here, as it could significantly impact the simulation model. Impact could reach the level of actually having to build a new simulation model and are therefore is not seen as an adjustment or scenario of the presented model. Furthermore, significant changes to the model make the difference in modelling structure and behavioral logic of the agents such that scenario-wise comparison with the calibrated based year state yields less robust results. In other words, significant changes to the model may require recalibration of the model.

4 Model user manual

This chapter describes how to run the python back-end simulation model. These instruction can also be found in the README.md which comes along with the simulation model. In this chapter examples of modelling output have been provided in graphs, the provided output is not based upon a specific model scenario and is only provided to give insight in the results the model directly produces.

4.1 System performance – Key Performance Indicators

To analyze the simulation model results a set of Key Performance Indicators (KPI) has been determined and is being calculated in the model. The technical description describes how a simulation result is being logged and KPI values are determined from the simulation log. This paragraph describes which performance indicators are currently produced by the simulation to give insight in the performance of the simulated scenario.

In theory the key element of the port nautical chain is that the pilot and tug services are provided to the ships who require these services and that the waiting times for the required services are minimal. The total duration of the nautical service chain for an incoming, departing or shifting ship gives insight in the performance of the total nautical service chain. The simulation results also allow to get more insight in which service provider has the most influence on this total service chain duration.

The general output of the model are global port averages for the KPIs listed above. The simulation produces data for these KPIs per ship movement, which in theory allows analysis of the performance on different levels of detail and different cross sections of the simulation output. However, due to data and model limitations the model has only been calibrated on global parameters and therefore the simulated results are most relevant on this aggregated level.

4.1.1 List of KPIs

1. Port performance
    a. Total turnaround-time port visit
    b. Sum of loss time caused by nautical chain processes
2. Performance of the port nautical chain
    a. Waiting times for nautical chain services
    b. Availability of services
    c. Occupancy of services

4.1.2 Definition of port performance KPIs

4.1.2.1 Total turnaround-time port visit

Average and standard deviation of total port visit duration. This KPI is determined by three key processes: number of shifts, time spend at berth and time spend in nautical chain processes. The latter is the process this simulation model focusses on and detailed insight in duration and delay in nautical chain processes can be found in detail KPIs below.

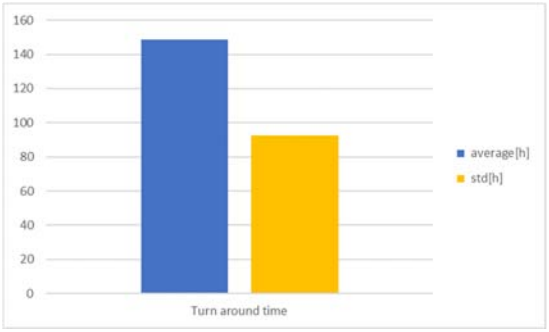

Figure 15: Example modelling output of average and deviation of turnaround time.

4.1.2.2 Sum of loss time caused by nautical chain process

In the simulation vessels spend time waiting for berth places or nautical chain services. This waiting time is seen as a loss and this indicator shows port performance on how much time is lost because of inefficiencies in port processes. Indicator is split in loss time caused by waiting for berth place and loss time caused by waiting for services.

4.1.3 Definition of nautical chain performance KPIs

4.1.3.1 Loss time in nautical chain processes

Duration of nautical chain processes compared to optimal nautical chain processes. Delay caused by waiting for services in nautical chain.
Average and Standard Deviation of the delay time per type of chain:

- Incoming;
- Shifting;
- Outgoing.

4.1.3.2 Waiting times for nautical chain services

Per type of nautical chain and per service average and standard deviation of the waiting time.

- Types of services
    - Berth;
    - Pilot;
    - Tugboat.
- Types of nautical chains:
    - Incoming;
    - Outgoing;
    - Shifting.

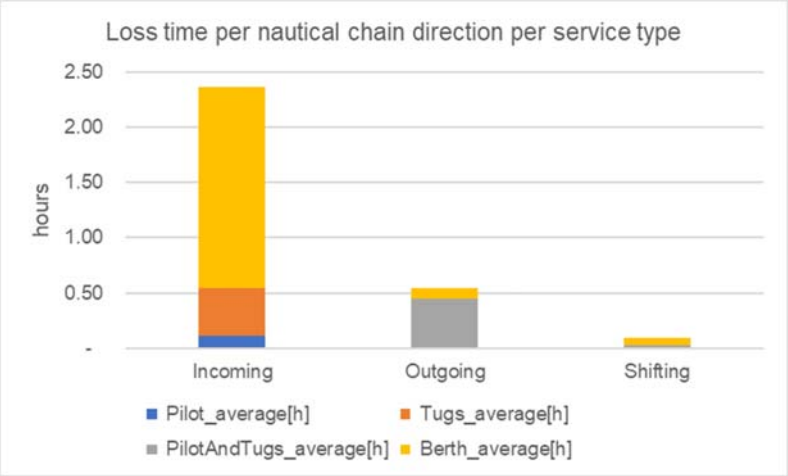

Figure 16: Example of modelling output average waiting per time per service type per direction of nautical chain. The sum of waiting times is the average of the sum of total loss time per nautical chain.

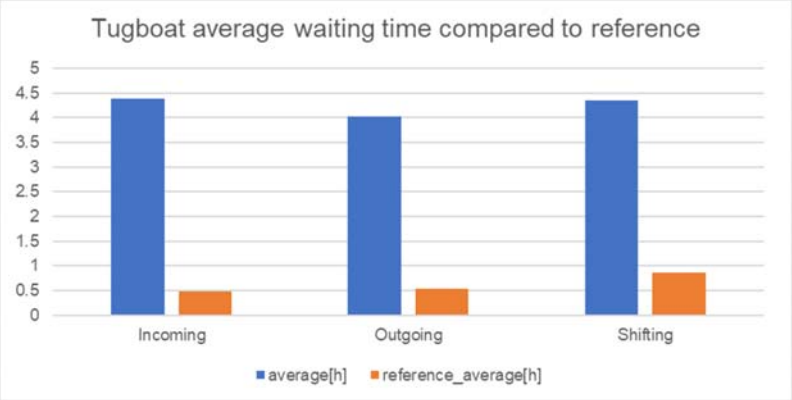

Figure 17: Example of modelling output average waiting time for tugboat service per direction in scenario with only 8 tugboats compared to the reference scenario with 20 tugboats.

4.1.3.3 *Availability of services*

This KPI indicates the direct availability for vessels that required service. In other words the share of vessels that get assigned a service immediately after requesting. In the simulation this is share is determined by counting the share of vessels that waits less then specific threshold time on services. Due to simulation processes for example a waiting time less than 15 minutes is seen as a direct availability of the service.

The share is given as a percentage per service type for these types:

- Berth;
- Pilot;
- Tugboat.

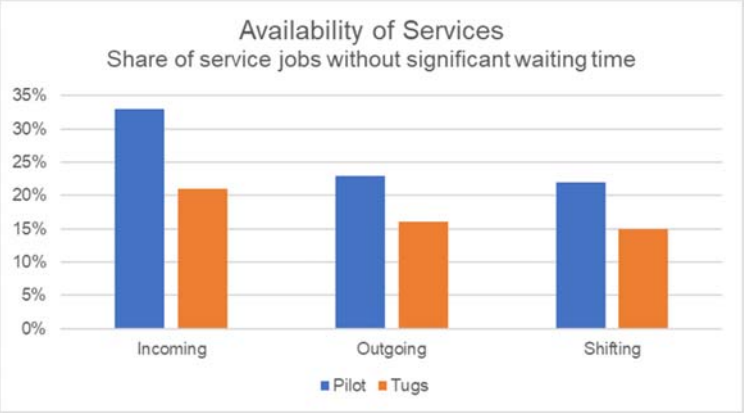

Figure 18: Example of modelling output availability of services in simulation run.

4.1.3.4 *Occupancy of services*

Indicating the occupancy of services is done by summing the amount of time services spend on:

- Idling
- Travelling to jobs
- Servicing jobs
- Waiting for other services

These indicators show how much of the capacity of the services is in use and what processes the services spend the most time on. Th above indicators are determined for pilots and tugboats.

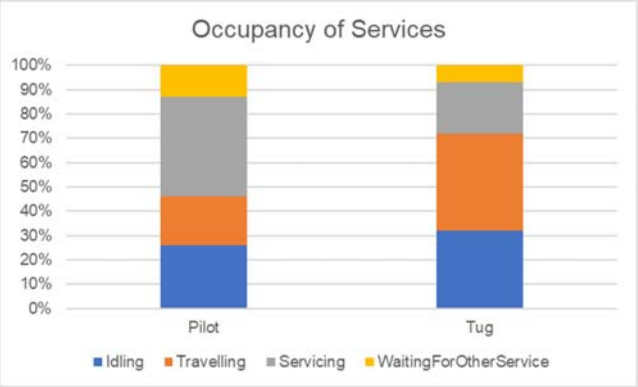

Figure 19: Example of modelling output occupancy of service agents in simulation run.

4.2 How to run the model

The simulation model can be activated running the script *RunSimulation.py* in for example a command window. Adding the argument '-t [nr of seconds]' changes the number of seconds the simulation will be running, differently from the default specified in the script.

The simulation will be run with the defined input files and parameters. Changing input parameters or files can be done to simulate various scenarios.

These are described in chapter 5 for scenario parameters and input files and in section 8.5.1 for simulation speed and resolution settings.

Each simulation run creates a new output folder where input and output of the simulation run are saved. Output files and application uses of the model are explained in the following two sections.

4.3 Output files

A simulation run with the model produces various output files. The different files have different uses from analyzing the simulation run in simulation log files until the performance output for scenario analysis. The different output files which can be found in the output folder of the simulation model are listed in the table below.

Output file	Description
simulation run output	
console_log.txt	logfile of console print statements, additional to simulation.log
df_for_stats.csv	table with details for df_stats.csv
df_ship_arrivals.csv	list of ship arrivals
df_start_variables.csv	parameters settings of simulation run
df_stats.csv	df_stats for comparing KPI and calibration
HAMIS_file.csv	log of HAMIS messages
kpi_overview.csv	table with KPI values of simulation run
numpy_seed.p	pickle file with numpy random seed for repeatings same simulation
parameters.py	copy of parameter input file
service_level.csv	table for service level analysis
simulation.log	logfile of all info, warning and error statements
VTS_file.csv	log of VTS messages
global simulation output	
last_simulation_dir.txt	list of simulation directories, last line is most recent
last_simulation_ids.txt	list of simulation ids, last line is most recent
console_logs	temporary copy of most recent log file

4.4 What-if analysis

The developed simulation is a scenario analysis tool where scenario's defined by varying input settings can be compared on various performance indicators, for example in comparison with the base year calibrated baseline performance.
The model can therefore also be considered as a 'what-if analysis' tool. Scenario changes can be made on various levels: on input parameters, input data or even agent behavior. Before changing any of the inputs and analyzing simulation results it is advised to get decent understanding of the simulation model and its capabilities. The more complex the scenario changes the more model understanding is required to ensure that requested changes are implemented correctly. Chapters 5, 6 and 7 go into detail on the different scenario changes which could be made, which analysis could be done with those changes and give insight in how to apply changes.

- Only parameter or data changes → chapter 5.
- Basic agent behavior changes → chapter 6.
- Agent interaction and communication changes → chapter 6. + 7.

4.4.1 How to run scenarios

This paragraph provides a short manual on what actions to undertake to run simulation model scenarios. The descriptions are quite general and most actions depend highly upon proposed scenarios. To effectively implement scenario changes to the model it is advised to first get a decent understanding of the agent-based simulation model.

Running scenarios and obtaining the result has been described in the following two steps in these subsections below:

1. Define and implement scenario changes and run simulation(s).
2. Analyse results

4.4.1.1 Define and implement scenario settings

1. Define scenario changes and how/where to implement in model parameters
    a. What changes to make in which input?
    b. How long or how often to run model / simulate?
        i. Determine the required sample size of number of nautical chains or number of ship visits to quantify scenario impact. With this number deduce number of days at least have to be simulated. Note that larger sample size and longer simulation runs will make simulated situation better approach to real world input distributions.
2. Implement changes in model input.
    a. Model parameters - SwarmportParameters.py
    b. Calibration parameters - CalibrationParameters.py
    c. Input data - /data
    d. Agent logic - [*agent script*]
3. Run scenario
    a. Single scenario with: RunSimulation.py
    b. Several scenarios with: GridSearch.py

4.4.1.2 Analyse scenario results

Results can be found in *'/simulation_output/[simulation_id]'*, the output folder per simulation run.

1. Analyze agent state durations to verify the success of the simulation run. First scenario impacts can be seen in changes in state durations. These results can be found in *df_stats.csv* and can be compared to the reference scenario and to real world state durations.
2. Analyze impact on performance indicators compared to baseline. The performance indicators of a simulation run can be found in kpi_overview.csv in the output folder.

5 How to edit scenario parameters

This chapter describes how to edit input parameters of the model, in other words to change the simulated scenario by variation in the input parameters or data. This means that changes can be made without changes in the software. Therefore no detailed knowledge of the model software is required to apply scenario changes with these parameters and subsequently evaluate scenario performance.

The scenario inputs of the model that can be changed are categorized in these four groups:

1. General scenario parameters
2. Agent speed parameters
3. Location parameters
4. Simulated situation

The first three categories are about changes in parameter values which are specified in the parameters input file of the simulation. Editing these values in this file creates a new scenario specific input file. The difference in these categories is the effect these parameters have on the simulated scenario.

The fourth category is about changing input data tables which describe the simulated Port of Rotterdam. Changes in these input data could increasing the size of the port or changing the distribution of ships over the different berth places.

Next to scenario input parameters there are also simulation parameters which can be altered in the input parameters, however they are not scenario parameters, as they do not influence the simulated scenario. They have an effect on the way the scenario is simulated. These parameters require more detailed technical understanding of the model and are therefore elaborated in section 8.5.1.

5.1 General scenario parameters

- Date of start of simulation
- Time resolution, how many minutes per simulation tick
- Number of pilot agents in simulation
- Number of tugboat agents in simulation
- Ship arrival rate in ships per day and type of stochastic distribution

5.1.1 Discrete time simulation

The method of simulation is based upon discrete time intervals at which at the end of each interval the state of all agents is updated given the time passed in the interval. This is implemented with a simulation ticker which informs all agents to update their status. The ticker counts how many time intervals have passed.

The interval size or time resolution can be altered as a scenario parameter which has impact on the speed and the accuracy of simulated results. A broader time interval speeds up the amount of simulated time, but it reduces result accuracy. As all process times of the various agent states get rounded to a whole number multiple of the time resolution. For example if the time resolution is set to one hour,

the agent can only change state once an hour and the time an agent spends in a certain state attains a minimum of one hour. Which is undesirable if the impact on nautical chain processing times on minute scale is of interest.

5.2 Agent speed parameters

The agent speed parameters, as the name states, determine the speed of the various agents in there different movement states. Changing the speed parameter of an agent has impact of processing times of the nautical chain.

These agent speeds are therefore also used as a calibration parameter to calibrate the baseline scenario of the simulation model to match processing times of elements the nautical chain to real world observations. To do this the agent speed parameters are a construct of a real world speed parameter and a calibration parameter. The real world speed parameter matches real world observed travelling speed. The calibration part of the parameter is used to calibrate the model and therefore this calibration parameter incorporates real world effects and system dynamics which are not represented in the ABSM. For scenario analysis purposes one should only edit agent real world speed parameters to match scenario needs and leave the calibration parameters untouched. Note that the model has been calibrated to a specific set of scenario parameters, significant changes to any of the scenario parameters might impact the quality of the calibration and simulation results, it is therefore advised to only analyze results of minor scenario parameter changes.

The speed parameter that can be changed in parameters file are listed below with additionally the processing times affected by these parameters.

1. Ship speed in port – unit is km/h
   Has a direct effect on the following process times
   a. Ship IncomingTugs
   b. Ship Outgoing
   c. Ship Shifting
   d. Pilot Incoming
   e. Pilot Shifting
   f. Pilot Outgoing
   g. Tug Tugging
2. Ship speed outside of port – unit is km/h
   Has a direct effect on the following process times
   a. Ship IncomingPilot
   b. Ship Outgoing
   c. Pilot Incoming
3. Pilot speed in port – unit is km/h
   Has a direct effect on the following process times
   a. Pilot ToTerminal
   b. Pilot Returning (to homeship)
4. Pilot speed outside of port – unit is km/h
   Has a direct effect on the following process times
   a. Pilot ToMaasCenter
   b. Pilot ToShipAnchorage
   c. Pilot Returning (to homeship)

5. Tug speed in port – unit is km/h

Has a direct effect on the following process times

a. ToStartPointTugging
b. Returning

5.3 Location parameters

5.3.1 The location parameter that can be changed in the model parameters are the resting locations of the pilots and tugboats.

5.3.2 Pilot

The pilot agent in the model has a resting location in the port of Rotterdam and a resting location on the open sea near Maascenter. This number of resting locations is fixed to these two locations in the current software setup. Scenario changes can be done in the location, coordinates, of these resting points. Which for pilots would change the response time from resting location to the stat location of service job.

The parameters that can be changed:

- Pilot boat location for available pilots at sea;
- Pistoolhaven for available pilots in port.

5.3.3 Tugboat

The tugboats have varying resting locations across the port, as can be found in the description of the tugboat allocation in paragraph 3.5.7. The number of resting locations and their coordinates can be altered to define new scenario setups of these resting locations. Just as with the pilots changing the location would mean a change in response time to the start location of service jobs.

5.3.4 Location and size of Anchor area

The anchorage area is specified by 4 input coordinates which are the corners of a 4-sided area.

5.4 Simulated situation

The developed simulation model is setup for the Port of Rotterdam. Therefore the situation or environment in which the agent-based simulation takes place is a representation of the Port of Rotterdam. The information making up this representation is contained in input tables, which can be altered for changes in port infrastructure or even be replaced for a whole new port.

This section describes how changes in this input can be made to edit the situation for example by adding a berth location of even a completely new terminal with several berths.

5.4.1 Berth and terminal locations

The input data for the berth locations also contains properties for the different berths. The properties describe the berth length, which determines the number of ships that can simultaneously berth, average berthing time and the visit frequency per berth.

These properties are used in the simulation to feed stochastic distributions that are used to determine agent parameters with model calibration, see chapter 7. The following parameters are drawn from stochastic distributions in the simulation for which the distribution is determined by data in one of the input files.

- Number of berth places in the port
- Number of times a berth is visited
- Time a ship spends at a berth

Berth locations and distribution of visits per berth can be found in this file in the simulation model folder: ‘data/berths.csv’.
This file contains:

- Berth name
- Terminal name
- Count of berth visits
- Cumulative count of berth visits
- Port section number
- Quay length in meters
- Average berth duration in hours from port call data
- Variance in berth duration in port call data

So changing the distribution of visits, or adding or removing a berth can be achieved by processing such a change in this input file. Note that if a new terminal is added with several berths that the terminal should also be added to the terminal file. The terminal file is located and named: ‘data/havens_latlon_routed_v2.csv’.
This file contains:

- Terminal name
- Port section number
- Route to this terminal from MaasCenter in port section numbers
- Coordinates of the terminal – projection WGS84
- Coordinates of the terminal – Mercator projection

More information on the coordinates and the used projections can be found in section 8.11.

5.4.2 Routing agents in the port

The terminal file also contains the sailing routes towards and from terminal locations. If changes in the routing of the agents are required than make sure update the route nodes file and the terminal file accordingly. An example of how this routing works can be found in Figure 20 below.

For each terminal the route of route nodes from port entrance to terminal location is listed. These route lists are also used to determine the routes for example for shifting vessels from terminal location A to terminal location B. This is done by reversing the entrance route of terminal A, sailing this route until the first node on the route to location B is crossed. From this point sail the route to B. This routing principle assumes the network of route nodes is a like a tree graph and there therefore always is a common node on routes from the start of the tree.

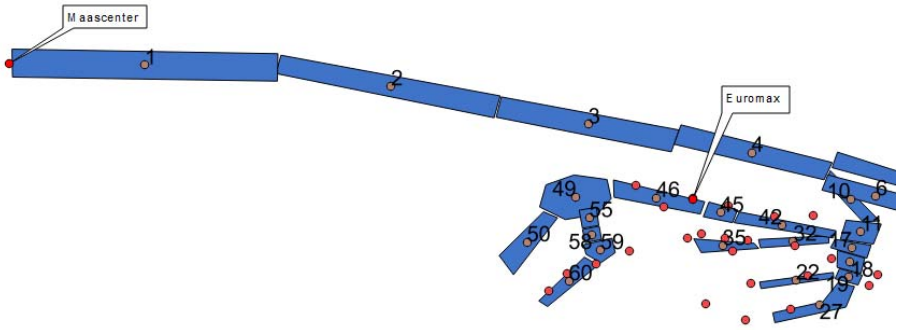

Figure 20: Schematic view of the Port of Rotterdam showing the port sections and numbering for the Maasvlakte. The port section numbers define the route to for example the Euromax terminal where the route is defined as the sequence of section numbers: 1, 2, 3, 4, 10, 11, 42, 45. Ships follow these route nodes to get to berths of this terminal and use the reversed sequence to find their way when departing the port from this terminal.

The port sections are a discretization of the routes ships sail through the Port of Rotterdam. This discretization is intertwined with the discrete time intervals in the simulation model. Therefore simulation model input on speed parameters and size of the discrete time intervals are also intertwined. What this means for the simulation model results and changing input parameters is discuses in the calibration in attachment B.

6 How to edit agent behavior

Changes in agent behavior in the simulation model can be required on different levels in the model. This chapter explains how to change elements on different levels for different agent types:

1. Physical agents
    a. Changing logic of existing state
    b. Adding a state to the state flow
2. Organizational agents
    a. Changing the service assignment scheduling method
    b. Adding a level of detail to service request and assignments

Note that the current setup of an agent-based model with a state machine and state flows interacting with states of agents is highly dependent on the fact that all state flows behave and react as intended. If one of the processes or elements in this flow is not working correctly that the whole simulation is disrupted, so all logic in the state flow model is crucial for a correctly operating simulation. Therefore it is advised to not change the agent logic without knowledge and deep understanding of all interactions and dependencies in the model.

One key difference between the physical agents and the organizational agents is that the physical agents have a speed and their location changes every tick of the simulation clock. The organizational agents do not have the requirement to work in synchronization with the global simulation ticker.

Extensive testing is required
When changing agent logic extensive testing is required. Stochasticity in the model implies that a wide range of simulation states are possible, Some problematic situations after agent changes only pop-up in very selective situations. This makes the testing process extensive to ensure changes are tested in all possible varieties of simulation states.

6.1 Physical agents – ship, pilot and tugboat

6.1.1 Adding logic to an existing state

These are the most basic changes in agent behavior in the model. These can be straightforward to implement if the required change only deals with agent only information or system wide information. In that situation, where information is already available, the changes do not require additional communication or information sharing and only imply changes in the logic in specific agent states.

Changing a state logic mostly deals with adding or changing conditions which determine when agents switch to a specific new state. The nautical chain interactions between the various agents causes that the change propagates to other agents by influencing the state changes for these other agents. If the state flow of a ship agent or of a service is changed, delayed or sped up, then it impacts the subsequent state flow of other agents.

6.1.2 Adding a new state

If the required agent behavioral change cannot be incorporated in existing agent state logic, then it is possible to add a new state to state flow of an agent. This paragraph describes the minimal changes to make for incorporation of a basic new state. The complexity of the required changes in combination with inter-relation between agent states can greatly increase the challenge of adding a new state.

For each case the steps described are the minimum of what is necessary to make a new state function properly within the agent architecture.

1. Develop functional description of the state and think through the following questions:
    a. Describe the new agent state, define the type of state: static or dynamic, movement or no movement. Depending upon agent state type a selection of (service) agent functions are initiated.
    b. What are the previous states from which the agent should move this new state?
        i. What are the conditional requirements for this state change to the new state?
    c. What are the following states in which the agent could move to after this state?
        i. What are the conditional requirements for the state change to this state from the new state?
    d. For each of the conditional requirements from questions above, what information is required and how can the agent obtain this information?
    e. Research which other agents rely/depend on of the states or state changes around this new state and verify that they would still function properly after adding this new state.
2. Define a new state name, e.g. "*NewAgentState*", note the CamelCase agent state naming in the agent info.
3. Develop agent function for new state with the same name in lowercase and underscores: *def_sts_new_agent_state(self).*
    a. Implement the logic described in the answer of question *1-c-i.* If conditions are satisfied change agent state.
    b. Make sure that there is logic implemented to make the agent flow to a new state from this function: *if condition: self.dfi['Sts'] = 'NextState'*
4. Update agent functions from the answer of question *1-b-i.* to ensure new agent state is integrated in agent flow.
    a. Make sure there is logic implemented that contains: *self.dfi['Sts'] = 'NewAgentState'*
5. Update the (service) agent state lists with this new state. E.g. if it is a movement state add it to the *self.MovementStates.*
    a. *self.MovementStates = ['NewAgentState', 'NextState', …]*

6.2 Organizational agents – Pilot organization, Tugboat organization, Port Authority

In the current agent-based model there are 3 organizational agents. They share the same properties and requirements as all 3 run parallel to the physical agents in the simulation. Their task is to pick up organizational requests from message boards(HAMIS) and return required organizational decisions to these message boards(HAMIS). Decisions made by these organizational agents impact the state flow the 'physical' agents which do run in synchronization with the global ticker.

6.2.1 Changing scheduling and assignment logic

In the implemented simulation model the tugboat and pilot organization both make use of the same, basic FCFS-assignment function. This function is implemented such a way that it can be replaced by a different function which executes a different method of scheduling.

Changing this function requires development of new scheduling function with a new scheduling method and subsequently the organizational agent should be adapted to use this new scheduling function.

There are the two general steps to take to implement such a change:

1. Develop new scheduling function.
2. Adapt organizational agent logic to use new scheduling function.

The results of the first step should be a python function with the following input and output, just as the currently implemented function:

- Input: available service agents, service requests
- Output: assignments of agents to requests, update of availability of service agents.

An idea for improvement is to transcribe the scheduling problem to a linear problem and use a linear solver to provide a (sub)-optimal solution – assignment allocation. Example of tools which can be used are Google OR-tools[2] or Gurobi[3].

Implementing optimization logic could lead to significant increase in calculation times for these service assignments. The schedulers are set up in a way that they do not have to work in synchronized way with the simulation. They are setup in parallel in their own threads, such that increase in calculation time for an assignment solution does not hinder the simulation. However, note that an important requirement for the calculation time is that the assignment solution is provided fast enough to the service agents and ship agents such that the nautical chain simulation is not delayed by the assignment calculation, since that would violate realism of the simulation.

6.2.2 Adding level of detail to assignment and scheduling

Next to changing the method of scheduling it could also be that more details are added to the scheduling problem. Adding conditions and requirements which also are taken into account in the real world scheduling problem could increase the real world representation of the model.

[2] https://developers.google.com/optimization
[3] https://www.gurobi.com/products/gurobi-optimizer/

Adding such details should be done if it is required to answer specific questions related to these additions with the simulation model.

Examples of increasing details are:

1. Include the different tugboat companies to the tugboat scheduling problem.
2. Include an experience level for pilots and only experienced pilots can pilot large vessels or vessels with dangerous goods.

Implementing the addition of a detail to the scheduling requires changes in several parts of the model. As the level of detail should be added to the ship agents with the service requests, the service agents being assigned to these requests and the scheduler itself.

The requirements for such a change are:

- Level of detail in
    - service requests and the service agents
    - adding a layer of detail to the vessel information and the service requests being send out.
- Level of detail in service agents and service agent availability information.
- Adding level of detail in the assignment/scheduling problem.

7 Model calibration and sensitivity analysis

This chapter gives a brief inside in the method of calibration and the sensitivity analysis that has been done with the model. Resulting conclusions are provided in below, details on the calibration can be found in the attachments of this report. Ideas for further research also resulted from the executed simulations and analysis and are also provided in this chapter.

The baseline calibration has been based upon port call data from the Port of Rotterdam for the years 2016 and 2017 and AIS data analysis [Kaljouw, 2019]. Details on the calibration of the model can be found in attachments A and B. These attachments describe the process of calibrating the current model version on the current data, whereas this chapter is meant as a guideline for recalibrating for a new baseline scenario or after model adjustments have been made.

Next to calibration sensitivity analysis has been executed to verify the model and to give insight in scenario possibilities and the results that can be observed from model output. This is described in attachment C.

Extensive description of calibration and sensitivity analysis in the following attachments:

A. Calibration method and reference scenario
B. Calibration results
C. Sensitivity analysis

7.1 Conclusions from calibration and sensitivity analysis

The main conclusion to be drawn from the calibration results and sensitivity analysis is that the port nautical chain simulation model is up and running and can be used for analysis on port nautical chain performance in various scenarios. However we note that model is a prototype and not a fully functional decision support system for the Port of Rotterdam.

The calibration and sensitivity analysis results show insight in limitations of the model, which off course are inherent to a model which is a limited representation of reality. However, it is good to know what the model can and cannot do and first ideas to extend the model applicability are provided in the further research below. More detailed conclusions on the model resulting from calibrating and analyzing the model are listed below.

7.1.1 Calibration

The goal of the calibration is to fit several simulation KPIs to real world KPIs a fit on multiple values is required. To score results of simulation runs with different parameter settings one uniform scoring value has been determined by summing the squares of the absolute error per KPI, similar to the technique of using the mean squared error (MSE). This sum of all squared errors indicates the quality of simulation results for the parameter set. The lowest sum indicates the parameter set which produced the results closest to the observed performance indicators.

The following input variables have been calibrated to reduce the MSE of the model output.

- Ship sailing speed in port and outside of the port
- Pilot transport speed in port and outside of the port
- Tugboat sailing speed in port and outside of the port

Next to input variables also the simulation resolution, the time step size of the simulation has been set. The so-called tick size influences the simulation speed but also the exactness of the results and process durations in the simulation. With all discrete process and agent interactions in the simulation the exactness of process durations influences overall model output.

- Choice of simulation resolution is a trade-off between simulation speed and exactness of simulation results. Tick size of 6 minutes has proven to provide a good balance.
- Calibration really improved fit of model results to real world observations. The sum of squared errors was reduced from a total of 53837 to 1313.
- Further improvement on calibration result would require calibration on additional parameters or settings or model improvements.
- Having 10 pilots in the simulation model with an arrival rate of 50 ships per day is a valid ratio of pilots to ships according to a nautical chain expert from the Port of Rotterdam.
- According to the same expert 20 tugboats in the simulation model is deemed a realistic number for the activities scope modelled. As in real world the tugboats may perform other tasks than pushing or towing deep sea vessels, an inclusion of those activities that do not directly relate to servicing deep sea vessel may increase comprehensiveness of the model.

7.1.2 Sensitivity Analysis

- A sanity check on number of ships visiting the port shows waiting time and loss times increase when there are more port calls. This shows a promising use case of model application for a deeper analysis that could lead to conclusions on the capacity limits of the port infrastructure and nautical services.
- Number pilots is a sensitive parameter and has significant impact on port turnaround time. Insufficient number of pilots leads to gridlocks related to the port's ability to handle deep sea vessels. The number of pilots in the system is therefore a critical capacity variable.
- Number of tugboats is a sensitive parameter too (in practice tugboats tend to be less of a bottleneck due to more spare capacity available), but changing the number tugboats in the system shows sensitivity of the system ability to handle deep sea vessels too with respect to the number of tugboats.

7.2 Further research

The model can further benefit from R&D work related to improvements of the modelling techniques, such as:

- Finer discretization of time resolution and more actions within a unit of time resolution (a tick). From the calibration insight has been achieved on which part of the processes could benefit of model improvement to make the model come closer to reality.

For example, looking into the delays caused by the discrete time steps and looking for a method to execute more actions within one time step.
- Overcoming computational challenge, as interpreting programming languages show their limits. The current method of agent-based modelling implemented in programming language Python requires a significant amount of computation time, speeding up simulation with another programming language or researching the delaying factors and investing on speed improvement in current implementation would benefit the model usability.

The model can be further improved by conducting more research in the following areas:

- Incorporate more agent-specific logic related to the scheduling processes, especially to the operations and organization of production processes of the pilots and tugboats. The current version of the model simplifies those processes in order to get practical and adequate results, however, a more realistic modelling of them would significantly benefit the quality of the model.
- In the sensitivity analysis only change of a single parameter has been analyzed. Where it could be interesting to analyze change of more parameters and make a 3d-mapping of the impact on simulation performance. This will result in a better understanding of the sensitivities within the model and interdependencies of functional aspects of the model.
- The effect of an empty port at the start of simulation influences model outcomes. If it is not taken into account, it can lead to skewed results. The effect of empty port could either be alleviated by letting the simulation start with a certain steady state situation. Further research is needed on the impact of system initiation and the most appropriate ways of doing it.
- Include more data or more advanced waiting statistics, including unfinished nautical chains. This will provide more clarity in the sensitivity analysis of the tugboats and ship arrivals.

8 Technical model description

This chapter deals with the technical implementation of the simulation model. The technical setup of the model is explained to give an insight in how the model has been setup. These descriptions are an addition to the basic technical details which are part of the GitLab repository.

The simulation model has been developed on a TNO GitLab repository. On this repository basic technical documentation, directly related to the code, has been added.

This basic documentation describes:

- different scripts and their function
- input files,
- input parameters,
- calibration parameters
- output files
- the different message structures

In this chapter reference links to parts of the TNO GitLab repository haven been placed, the repository contains the most up to date versions of the technical details. This report a copy of this the technical details for the model version of July 2020.

8.1 Model architecture

8.1.1 Overview of scripts and parallel instances

The agent-based model described in chapter 3 has been implemented in python 3. The simulation model consists out of number of scripts each with its own functionality within the simulation. For example, for each type of agent there is a script that contains the class definition of the agent and all required functionalities in the class definition.

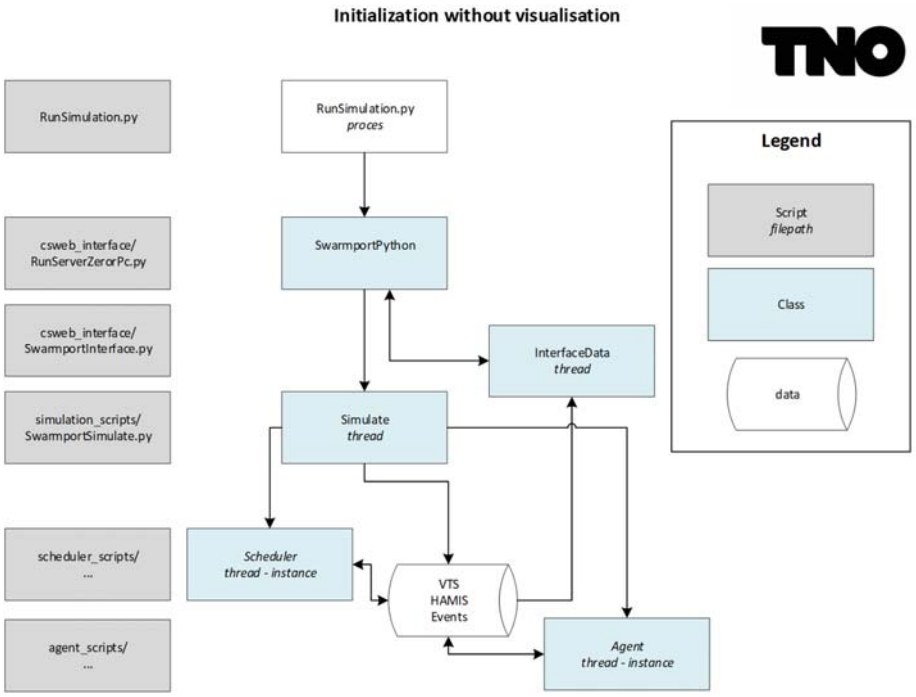

Figure 21: Overview initialization flow through the of scripts.

8.1.2 *An overview of all the different scripts and their functionality.*

Modules	Subfolder	Comment
RunSimulation		Script to startup back-end simulation without CsPort visualization framework.
SwarmportParameters		Script, file that contains all parameter settings for running a simulation
File location		Script, file that contains references to file paths
SwarmportSimulate	Simulation_scripts	Class defining simulation step, activating agent procedures.
SwarmportFunctions	Simulation_scripts	Class containing basic functions, such as routing functions, ...
SwarmportAgent	Agent_scripts	Parent class for all agents
SwarmportShip	Agent_scripts	Class defining agent behavior for each agent status
SwarmportServiceAgent	Agent_scripts	Parent class for Pilot and Tugboat agents
SwarmportPilot	Agent_scripts	Class defining agent behavior for each agent status
SwarmportTugboat	Agent_scripts	Class defining agent behavior for each agent status
SwarmportScheduler	Scheduler_scripts	Class defining basis scheduling functions for pilot and tug scheduler
SwarmportPilotScheduler	Scheduler_scripts	Class defining scheduling functions for tug scheduler
SwarmportTugboat Scheduler	Scheduler_scripts	Class defining scheduling functions for pilot scheduler

8.1.3 Class inheritance

Ship, Pilot and Tugboat all inherit agent functionalities from a parent class 'Agent'. Where the Tugboat and Pilot also inherit service agent functionalities from the parent class 'Service Agent'. This is done such that changes in basic agent structure only have to done in a single script to have effect on all different agents in the simulation model.

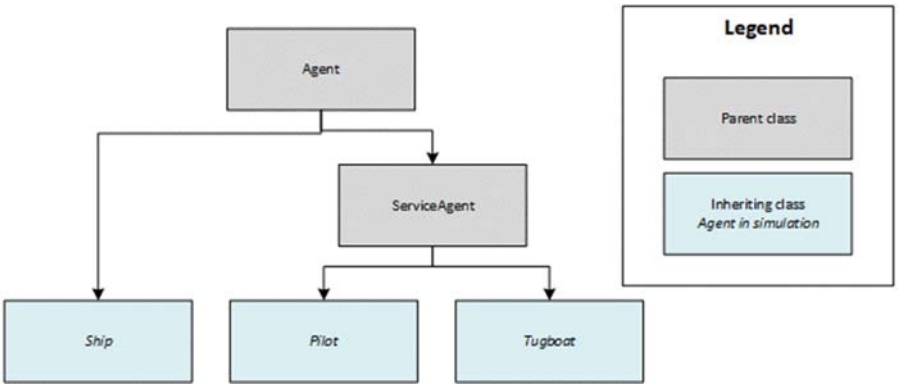

Figure 22: Flow of inheritance from parent class Agent to subclasses.

In similar manner the scheduler classes, PilotScheduler, TugboatScheduler and PortAuthority also share the same parent class ParentScheduler.

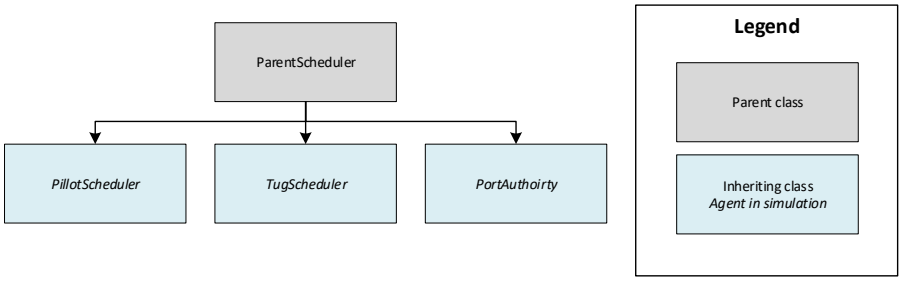

Figure 23: Flow of inheritance from parent class ParentScheduler to subclasses.

8.1.4 Name and coding conventions

Name example	***General use***
Self.variable_name	Class parameter or variable used in class or subclasses
Self._function_name	Class or agent functions used within class and subclasses
Self._sts_function_name	Agent function containing state logic of agent, used within class and subclasses
Logger.error(*statement*)	Printing statement to logfile of logger on all logging levels, see logconfig.py
Logger.warning(*statement*)	Printing statement to logfile of logger on warning logging level, see logconfig.py
Logger.info(*statement*)	Printing statement to logfile of logger on info logging level, see logconfig.py
Logger.debug(*statement*)	Printing statement to logfile of logger on debug logging level, see logconfig.py

8.2 Model parameters

Parameter		Description
Simulation settings		
RealWorldStartDatetime	'31-12-2016'	Start time and date of simulation
MaxShipsInSimulation		Limit number of ships to ensure stable simulation
MinutesPerTick		Time resolution of simulation in minutes per tick
SimulationDays		Number of days to be simulated
use_numpy_seed	True/False	Use random seed of previous simulation
Calibration parameters		
NumberPilots		Nr of Pilots in simulation
NumberTugboats		Nr of Tugs in simulation
ShipArrivalRate		Arrival rate in ships per day for default poisson distribution
modification_factor_ship_speed_at_sea_kmh		calibration factor for speed at sea
modification_factor_ship_speed_in_port_kmh		calibration factor for speed in port
ship_distance_at_sea_factor		distance to destination for state change
ship_distance_in_port_factor		distance to destination for state change
modification_factor_pilot_speed_at_sea_kmh		calibration factor for speed at sea
modification_factor_pilot_speed_in_port_kmh		calibration factor for speed in port
pilot_distance_at_sea_factor		distance to destination for state change
pilot_distance_in_port_factor		distance to destination for state change
modification_factor_tug_speed_at_sea_kmh		calibration factor for speed at sea
modification_factor_tug_speed_in_port_kmh		calibration factor for speed in port
tug_distance_at_sea_factor		distance to destination for state change
tug_distance_in_port_factor		distance to destination for state change
Model parameters		
request_hours_before_departure		Hours before service request departure
ship_speed_at_sea_kmh		ship speed at sea (without calibration)
ship_speed_in_port_kmh		ship speed in port (without calibration)
speed_at_sea_tug_kmh		tug speed at sea (without calibration)

speed_in_port_tug_kmh	tug speed in port (without calibration)
pilot_speed_at_sea_kmh	pilot speed at sea (without calibration)
pilot_speed_at_land_kmh	pilot speed in port (without calibration)
coordsStartShip_wgs	Location of ship entering simulation
coordsEndShip_wgs	Location of ship leaving simulation
coordsPistoolhaven_wgs	Location of pilot rest location in port
coordsHomeship_wgs	Location of pilot rest location at sea
anchorArea1_wgs	Polygon that describes anchor area
area_tugs_required_maasvlakte_lon_WGS	Longitude at which vessels approaching maasvlakte require tugboats
area_tugs_required_centrum_lon_WGS	Longitude at which vessels approaching Maascenter require tugboats
dict_tugboat_resting_locations	Resting location for tugboats
start_tugging_maasvlakte	section number of start tugging to maasvlakte
start_tugging_other	section number of start tugging
dict_start_point_tugging_locations	routes to start tugging locations
Model parameter files	
data/df_terminals.csv	table of all simulated terminals and their info
data/berths.csv	table of all simulated berths and their info
data/KPI_from_data_to_csv.csv	Observed KPI values from port call data

8.3 Input data

This paragraph lists the specific datafiles which are input files for the simulation. Use and changing of these files are described in chapter 5.4

Input file	Source	description
data/havens_latlon_routed_v2.csv	Port of Rotterdam	Location of and routes to terminals
data/berths.csv	Port of Rotterdam	berths and their specifications
shapefiles/sections_centroids.shp	TNO & Port of Rotterdam	Location of centroids of port sections
KPI_from_data.csv	TNO	Observed KPI values for calibration

8.4 Simulation output

Output file	description
simulation run output	
console_log.txt	logfile of console print statements, additional to simulation.log
df_for_stats.csv	csv ';' with details for df_stats.csv
df_ship_arrivals.csv	list of ship arrivals
df_start_variables.csv	parameters settings of simulation run
df_stats.csv	df_stats for comparing KPI and calibration
HAMIS_file.csv	log of HAMIS messages
numpy_seed.p	pickle file with numpy random seed for repeating same simulation
parameters.py	copy of parameter input file
simulation.log	logfile of all info, warning and error statements
VTS_file.csv	log of VTS messages
global simulation output	
last_simulation_dir.txt	list of simulation directories, last line is most recent
last_simulation_ids.txt	list of simulation ids, last line is most recent
console_logs	temporary copy of most recent log file

8.4.1 Simulation logging for KPI calculation

KPIs in the simulation are collected on 2 different ways. The KPIs of the simulation mainly focus or are based upon the time different agents are in different parts of the processes. E.g. the average time a vessel has to wait for pilot service.

The states of the agents are continuously being logged in the VTS message platform in the simulation. From the log of this platform the time per state per agent is defined at the end of the simulation. In this way only process times of individual agents are being logged. It could be that for other KPIs or required simulation results different process times would have to be logged to allow simulation performance calculation. For these results which cannot be attained from the VTS log it is possible to use the *self.dict_event_log* dictionary of the ship agent. For now this dictionary logs contain no additional information compared to the VTS, in the future they could be used for specific performance indicators.

8.4.2 Simulation statistics

From this logging a general overview of agent status duration is being produced in the file *'df_stats.csv'*. This file shows per agent type and per status the number of occurrences in a simulation and the descriptives on the status duration. From these descriptives the quality of the simulation run can be verified. In this file it is good to check the number of occurrences of agent states, the average time spend in each agent state and the deviation throughout the simulation. Note minor variation is possible due to a cut off of some processes at the end of simulation or minor deviations in log files or analyses of these log files.

- The number of occurrences should be in a similar order for the parallel processes of the different agents in the nautical chain. For example the number of outgoing ships should match the number of outgoing pilots.

- Average time spend in agent states should match real world nautical chain processes. For same agent duration the real world observed value is also given in this file, this is used for calibration purposes.
- Deviation in agent states duration should stay in natural boundaries. High deviations, e.g. standard deviation is higher than average, can indicate some processes in simulation do not run smoothly and most likely not as intended to be.

8.4.3 KPI calculation – nautical chain service level

The simulation calculates the key performance indicators which are described in section 4.1. A simulation run logs all vessel and agent operations in the simulation output and from this output the performance indicators are calculated and written in the KPI-overview file. Table 1 shows the KPI value resulting from the simulation.

Table 1: Example of KPI tables in simulation output. These numbers are for to provide insight in structure and content of model output and not based on a specific scenario.

Turnaround times:	
count	585
average[h]	58.24
std[h]	48.79

Direction	**total[h]**	**count_waiting**	**count_total**	**average[h]**	**median[h]**	**std[h]**
Incoming	9820.2	585	719	16.79	8.00	20.16
Outgoing	984	585	719	1.68	1.00	1.42
Shifting	5275.9	324	7909	16.28	9.75	17.77

Service levels - availability of services		
Pilot - Direct availability of service		
	Incoming	99%
	Outgoing	92%
	Shifting	70%
Tugs - Direct availability of service		
	Incoming	66%
	Outgoing	64%
	Shifting	25%
Berth - Direct availability of service		
	Incoming	25%
	Outgoing	68%
	Shifting	23%

Service	**Direction**	**total[h]**	**count**	**average[h]**	**median[h]**	**std[h]**
Pilot	Incoming	74.3	585	0.13	0.10	0.08
Tugs	Incoming	250.1	585	0.43	0.30	0.36
Berth	Incoming	9495.8	585	16.23	7.50	20.17
Pilot	Outgoing	113.1	585	0.19	0.10	0.21
Tugs	Outgoing	302.9	585	0.52	0.20	0.54
PilotAndTugs	Outgoing	302.9	585	0.52	0.20	0.54
Berth	Outgoing	265.1	585	0.45	0.30	0.39
Pilot	Shifting	140.8	324	0.43	0.40	0.30
Tugs	Shifting	303.5	324	0.94	0.85	0.61

PilotAndTugs	Shifting	303.5	324	0.94	0.85	0.61
Berth	Shifting	4528.1	324	13.98	6.85	17.41

Occupancy per service				
Pilots				
Idling	Travelling	Servicing	WaitingForOtherService	
37%	17%	39%	7%	
Tugboats				
Idling	Travelling	Servicing	WaitingForOtherService	
43%	34%	18%	5%	

8.5 Simulation core

The heart of the simulation can be found in the script *SwarmportSimulate.py*. This script, in more detail the function *Simulate.run()*, contains the logic that makes the simulation tick. This function makes the simulation iterate through time and in each iteration activate the agents and wait for the agents to complete their actions. In each tick different agent states are subsequently activated, this is described in section 8.5.2. The speed of simulation can be influenced by the simulation parameters as described in the paragraph below.

8.5.1 Simulation parameters

There are several input parameters that influence the speed and time resolution of the simulation. All parameters are described in paragraph 8.2. A more detailed explanation of simulation speed parameters is given below.

Simulation speed, the amount of time requirement to simulate a period of real world time can be decreased by simulating with bigger time steps, see also paragraph 5.1.1. The downside of this is that a lot of simulation result details are lost, because the processing times of agent states would get logged on this bigger time interval.

Also there is the agent event time out delay, which is the maximum time the simulator waits for agents to handle their actions. This feature provides a safety warning for model users when there is something going wrong in the simulation. Agents which subsequently do not succeed to finish processing in this event time, will start to ‘run behind’ on the simulation. A warning has been built in to notify the user if this problem occurs, if this situation occurs then the warning will be printed in the console where the simulation model has been activated. A possible solution would be to solve the agent processing speed or to increase the time out delay and slow down the simulation.

8.5.2 Description of agent state machine flow

Each simulation iteration, in this report referred to as ‘tick’, agents are activated. Activating the agents has been split up in three steps. This has been done to ensure all agents can activate there logic accordingly, because in the agent logic there are many dependencies on states and locations of other agents. Therefore, firstly a separation has been made to movement action and check status action. Secondly the service agents have states in which they ‘follow’ the location of the ship agents, for these states the move-along step has been introduced. This step ensures correct movement to the new location of the vessel who updates its location in the movement action before the move along step.

The third and last step is the check status action where all agents check the conditions of the state flow for changing to a next agent state, as described in the agent state flows in chapter 3.

Serial process flow for physical agents: ships, pilots and tugboats:
1. Movement actions (movement states)
2. Move along (move-along states)
3. Check status (all states)

Parallel process flow for other agents: port authority, pilot company, tugboat company.

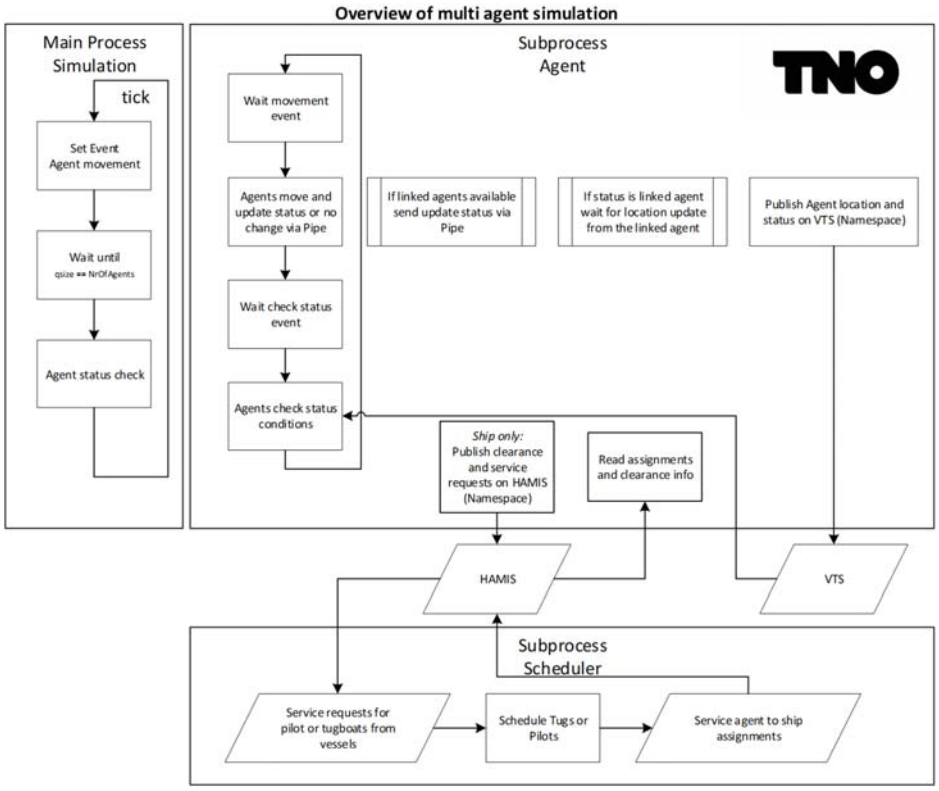

Figure 24: Overview of processes in the multi agent simulation model.

8.6 Connection to visualization framework

The focus of development of the agent-based simulation model was on the agent-based model. More specifically, in modelling and calibrating the agent behavior and agent interaction to make the simulated nautical chain process a representation of the real world nautical chain. At the start of the project a visualization module was made to support model development, model validation and dissemination. However, this module is no use for the long term strategic simulation and calculation of the KPIs, which is the purpose of the model. Therefore the visualization module is no core element of the model and only described in this section of this report.

As the visualization software developed at the start of the project was built upon no longer supported software. This section mainly focusses on how to connect a possible new module to the simulation. A short description of the developed model with TNO Common Sense is given as well.

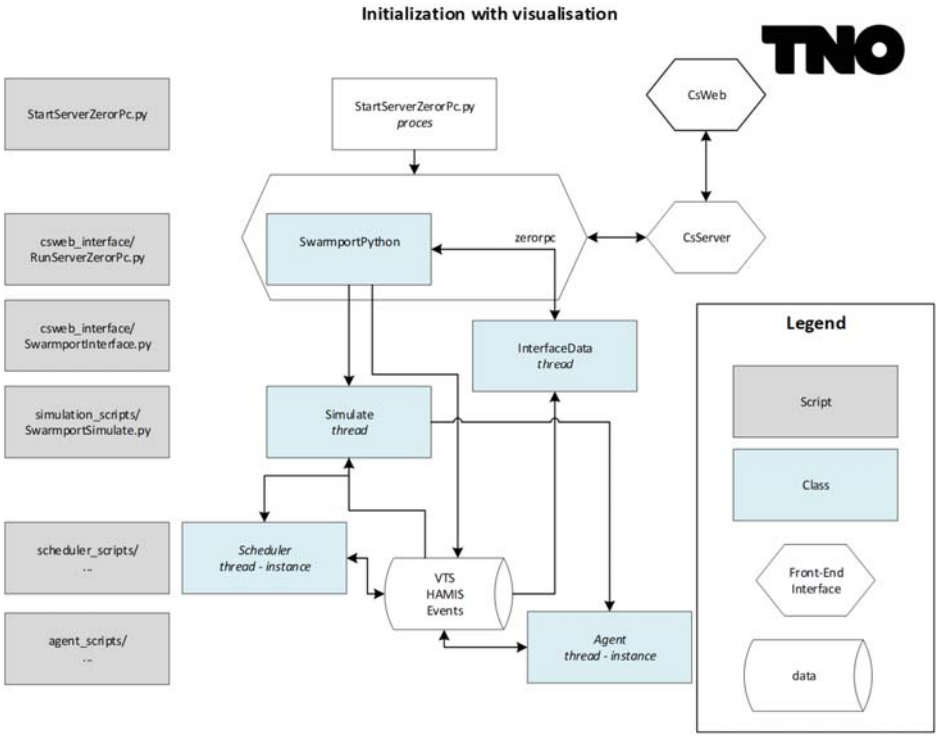

Figure 25: Simulation model architecture with connection to the CsPort module by using the zerorpc connection software.

8.6.1 CsPort – Common Sense Swarmport Module

For visual validation and dissemination, it is possible to visualize the simulation with a Common Sense module, build on TNO development open source software CsWeb [CsWeb]. However the at the start of project constructed module for visualization is based upon a version of CsWeb which is no longer in use at the time of writing this report.

8.6.2 Connection setup between python and node.js

- Zerorpc python package
- SwarmportPython.py simulation model connection-end
    - Simulation controls

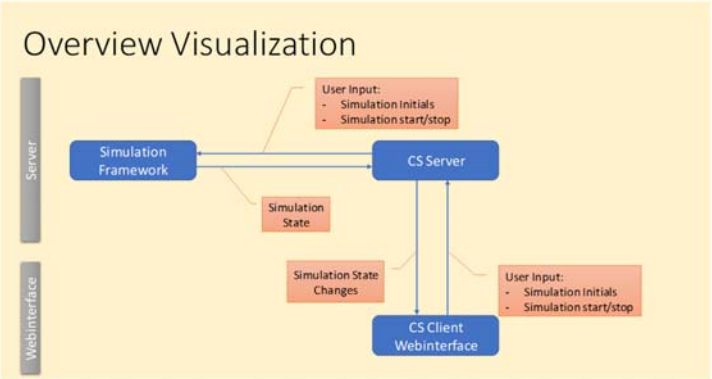

Figure 26: Overview of setup of the visualization

8.6.3 Connection setup with other visualization tool

- Method required for connecting SwarmportPython.py to the visualization tool
  - Setting up a connection where the used visualization tools can activate the Swarmport simulation controls and receive simulation information via a data connection.

8.7 Status structure agents

In the functional description in chapter 3 the processes per agent type are described. Due to implementation requirements some differences have been made in state structure or state naming in the simulation model. The model is still the same functional model as described before, but only to ensure smooth simulation and being able to achieve required performance indicators some additional agent states have been implemented. The table below relates the agent states in the model with their functionalities as described in chapter 3.

Agent	State - implemented	State - functional (chapter 3)
Ship		
	initialized	PreBirth
	sailing_to_maas_center_buoy	SailingToMaasCenterBuoy
	incoming_anchorage	IncomingAnchorage
	at_achorage	AtAnchorage
	incoming_pilot	IncomingPilot
	waiting_intoming_pilot	WaitingPilot
	incoming_tugs	IncomingTugs
	waiting_tugs	WaitingTugs
	incoming	Incoming
	at_terminal	AtTerminal
	preparing_for_shifting	CallShipAgent
	awaiting_services_shifting	PrepareForShifting
	shifting	Shifting
	preparing_for_outgoing	CallShipAgent

	awaiting_services_shifting	PrepareForOutgoing
	outgoing	Outgoing
	out_of_port	OutOfPort
	dead	OutOfPort
Pilot		
	available	Available
	assigned	Assigned
	to_maas_center	ToMaasCenter
	await_ship_incoming	ToShipIncoming
	to_terminal	ToTerminal
	ready_for_outgoing	ToTerminal
	ready_for_shifting	ToTerminal
	incoming	Incoming
	shifting	Shifting
	outgoing	Outgoing
	returning	Returning
	transfer_to_homeship	Transfer
	transfer_to_pistoolhaven	Transfer
Tugboat		
	available	Available
	assigned	Assigned
	to_startpoint_tugging	ToStartPointTugging
	ready_for_tugging_outgoing	ToShip
	ready_for_tugging_shifting	ToShip
	await_ship_incoming	ToShip
	tugging_incoming	Tugging
	tugging_shifting	Tugging
	tugging_outgoing	Tugging
	returning	Returning
	returning_but_assigned	Returning

8.8 Data structure agents

Agent info is stored in attribute self.dfi of each agent-thread. Following basis dfi structure for all agents and further below agentspecific adaptions. Where dfi is a Pandas Series and contains the following elements:

dfi["AC"] = AgentClass
dfi["AT"] = AgentType
dfi["CC"] = CurrentCoords
dfi["CD"] = CoordsDestination
dfi["He"] = Heading
dfi["Sts"] = Status

dfi["TD"] = Task Destination
dfi["DD"] = Distance to destination

8.8.1 Additional data structure for ships

dfi["RemainingTerminals"] = List of remaining terminals that need to be visited
dfi["TD"] = Task Destination: terminal
dfi["TC"] = Terminal currently visiting (or most recently visited)
dfi["TD_berth"] = Task Destination: berth
dfi["TC_berth"] = berth currently visiting (or most recently visited)
dfi["RemainingCargo"] = List of cargo per terminals that needs to be visited
dfi["Cargo"] = Cargo for upcoming terminal
dfi["DD"] = Distance to destination
dfi["Clearance"] = Clearance for entering, shifting or leaving port
dfi["PilotAssigned"] = Pilot for entering / leaving port assigned
dfi["TugsAssigned"] = Tugboats for entering / leaving port assigned
dfi["Direction"] = Direction of nautical chain incoming, shifting or outgoing

8.8.2 Additional data structure for pilots

dfi["Direction"] = Direction of nautical chain incoming or outgoing

8.8.3 Additional data structure for tugboats

dfi["Company"] = tugboat company – only 1 type in current setup

8.9 Communication structure

8.9.1 Methods of inter-thread communication

By using python multiprocesing Namespace(), Pipe(), Queue() and Event()

1. Namespace is used for general published messages - location on VTS and port call and service requests on HAMIS
2. Pipe is used for 1 - 1 agent communication, for service agent to move along ship
3. Queue is used for event structure and for service requests queuing.
4. Events for event structure to trigger events among all processes

8.9.2 Basic agent event structure

1. Wait for event triggers movement or status
2. Trigger event update movement or status and
   – send 'done' to Queue() - sim_queue
   – send status/movement update to VTS

8.9.3 Ship communication structure

1. Post service request on HAMIS and in pilot_requests_queue en tugs_requests_queue
2. Listen to HAMIS for pilot and tugboat assignments
3. Post location update via ship_side_pipes to all assigned agents

8.9.4 Pilot and tug communication structure

1. Listen to HAMIS for assignments
2. Listen to pilot_side_pipe for location and status updates of ship

8.10 Message structures

These message structures are critically implemented in the following scripts:

- agent_scripts\Agent.py
- agent_scripts\ServiceAgent.py
- agent_scripts\SwarmportShip.py
- scheduler_scripts\SwarmportParentScheduler.py

8.10.1 VTS – Vessel Traffic management System

8.10.1.1 Agent to VTS

Dfi as Pandas Series as specified before.

```
setattr(self.VTS, agent_name, self.dfi)
```

8.10.1.2 VTS to Agent

Dfi as Pandas Series as specified before.

```
agent_dfi = getattr(self.VTS, agent_name)
```

8.10.2 HAMIS

8.10.2.1 Agent to HAMIS

Agent putting msg_HAMIS into to HAMIS

```
setattr(self.HAMIS, agent_name, message_dict)
```

Agent class initialization

```
msg_HAMIS = {}
```

Service agent subclass initialization

```
msg_HAMIS = {'ship_side_pipe': self.ship_side_pipe}
```

8.10.2.2 Ship to HAMIS

Requesting clearance

```
msg = {
        'type': 'call_for_clearance',
        'id': self.name,
        'status': self.dfi['Sts'],
        # 'ETA': self.dfi['ETA'],
        'terminal': self.dfi['TD'],
      }
setattr(self.HAMIS, f'call_for_clearance_{self.name}', msg)
```

Requesting pilot

```
msg = {
        'type': 'call_for_pilot',
        # 'ETA': self.dfi['ETA'],
```

```
                'id': self.name,
                'terminal': self.dfi['TD'],
                'origin': [terminal / maascenter],
                'destination': [terminal / maascenter],
                'cargo': '#TODO'
            }
        setattr(self.HAMIS, f'call_for_pilot_{self.name}', msg)
```

Requesting tugboats

```
    msg = {
            'type': 'call_for_tugs',
            # 'ETA': self.dfi['ETA'],
            'id': self.name,
            'terminal': self.dfi['TD'],
            'origin': [terminal / maascenter],
            'destination': [terminal / maascenter],
            'cargo': '#TODO',
            'nr_of_tugs': nr_of_tugs
        }
    setattr(self.HAMIS, f'call_for_tugs_{self.name}', msg)
```

8.10.2.3 Scheduler to HAMIS

Giving clearance to ship

```
setattr(self.HAMIS, f'clearance_granted_{ship_name}', True)
```

Assigning pilot / tugboat to ship (message for service agent)

```
service_agent_msg = {
    'type':'assignment_info',
    'origin': [terminal / maascenter],
    'destination': [terminal / maascenter],
    'assignment_type': 'Incoming' or 'Shifting' or 'Outgoing',
    'assignment_id': ship_id,
    'ETD': ...,
    'ETA': ...,
}
setattr(self.HAMIS, f'assignment_{service_agent}', service_agent_msg)
```

Assigning pilot to ship (message for ship)

```
ship_msg = {
    'ids': ['p1'],
    'ETA': ...,
    'ETD': ...,
}
setattr(self.HAMIS, f'assigned_pilot_to_{ship}', ship_msg)
```

Assigning tugboats to ship (message for ship):

```
ship_msg = {
    'ids': ['t1', 't2'],
    'ETA': ...,
    'ETD': ...,
}
setattr(self.HAMIS, f'assigned_tugs_to_{ship}', ship_msg)
```

8.10.3 Pipe

8.10.3.1 Pipe message structures

These message structures are critically implemented in the following scripts:

- agent_scripts\ServiceAgent.py
- agent_scripts\SwarmportShip.py

8.10.3.2 Ship in pipe to service agent

Currently, the pipes are one-way communication from the ship to the service agents. The ship communicates two messages: (1) his updated location, so that the service agents may move along, and (2) a flag that he updated his status, since the status update of the service agents may depend on the current status of the ship.

1. Location update:

```
msg_service_agents = {
    "time": self.VTS.clock,
    "msg_type": "location_update",
    "CC": self.dfi["CC"],
    "DD": self.dfi["DD"]}
ship_side_pipe.send(msg)
```

2. Status update:

```
msg2 = {"time": self.VTS.clock,
        "msg_type": "status_update"}
ship_side_pipe.send(msg2)
```

8.10.4 Queues

These message structures are critically implemented in the following scripts:

- agent_scripts\ServiceAgent.py (sim_queue only)
- agent_scripts\SwarmportShip.py
- simulation_scripts\SwarmportParentScheduler.py

8.10.4.1 Ship to PilotScheduler

From to ship to pilot_requests_queue:

```
self.request_pilot_queue.put({'id': self.name})
```

8.10.4.2 Ship to TugScheduler

From to ship to tugs_requests_queue - several requests to the queue, one for each required tugboat:

```
for _ in range(nr_of_tugs):
    self.request_tug_queue.put({'id': self.name})
```

8.10.4.3 Request clearance queue

From ship to scheduler

```
msg_HAMIS = {
    'type': 'call_for_clearance',
```

```
    'id': self.name,
    'status': self.dfi['Sts'],
    # 'ETA': self.dfi['ETA'],
    'terminal': self.dfi['TD'],
}
self.request_clearance_queue.put(msg_HAMIS)
```

8.10.4.4 Request pilots queue

From ship to PilotScheduler:

```
self.request_pilots_queue.put({'id': self.name})
```

8.10.4.5 Request tugs queue

From ship to TugScheduler - several requests to the queue, one for each required tugboat:

```
for i in range(nr_of_tugs):
    self.request_tugs_queue.put({'id': self.name})
```

8.10.4.6 Simulator queue

From all agents to simulator. Used for synchronising agents (wait for message from every agent, then initiate new event):

```
if self.dfi['LastUpdate'] == self.VTS.clock:
    msg_scheduler = {'update': self.dfi, 'type': 'status',
                     'id': self.name}
else:
    msg_scheduler = {'nochange': {}, 'type': 'status',
                     'id': self.name}
self.sim_queue.put(msg_scheduler)
```

8.11 Coordinates and global projection

The simulation model simulates the movement of agents throught the Port of Rotterdam. The location of the agents and physical locations in the simulation are defined in coordinate system known as the Mercator projection. This projection has been chosen because it allows agent to sail to another set of coordinates with a constant bearing.

To be able to work with input data which is based on the commonly used WGS84 projection coordinate transformation was needed.

This has been done using the PyProj python package. Using the transform function with the following settings:

- Mercator projection: init=‘epsg:4326’
- WGS84: init=’epsg:3857’

9 References

Davydenko et al., 2019 — I. Y. Davydenko, R.W. Fransen, *Conceptual Agent-based Model Simulation for the Port Nautical Services*, 15th IFAC Symposium on Large Scale Complex Systems, Delft, 2019.

Kaljouw, 2019 — S. Kaljouw, *Tugboat resting location optimization using AIS data analysis*, Erasmus University Rotterdam, 2019.

Port Information Guide — Port of Rotterdam, *Port Information Guide*, April 2020, https://www.portofrotterdam.com/sites/default/files/port-information-guide.pdf, accessed 6 April 2020.

Verduijn, 2018 — A.M. Verduijn, *Identifying the relations between and mapping the processes of the nautical service providers in the Port of Rotterdam*, Master Thesis, TU Delft, 2018.

10 List of abbreviations

ABSM	Agent-based Simulation Model
AIS	Automatic Identification System
ERP	Early Research Program
KPI	Key Performance Indicator
NWO	Nederlandse organisatie voor Wetenschappelijk Onderzoek

11 Signature

The Hague, 21 January 2021

TNO

Paul Tilanus
Projectleader

Ruben Fransen
Author

A Calibration method

This attachment describes the reference scenario and the calibration method. How a set of parameters was defined that make the simulation model such that the indicators from Table 2 are reproduced. At first the reference scenario is described. Secondly, the effect of a change of the input parameters on the performance indicators is sketched. This is done to provide some insight into the calibration process.

A.1 Reference scenario

The Port of Rotterdam has provided data on the port nautical chain operations and from the AIS[4] system data is available containing real world observations of ship movements. From these real world data a number of performance indicators can be computed to quantify the performance of the port nautical chain.

The implementation of a simulation model for the port nautical is a simplification of the port with a small number of parameters. To provide an accurate description of the actual port, it would be convenient if the parameters can be chosen in such a way that the performance indicators of the simulation correspond to those from the observation data. So, there are two descriptions of the port nautical chain and the goal of the calibration is to bring them together.

The performance indicators used are the most significant indicators that are both known and clearly defined in both the observation data and the simulation model. This means that these indicators can be described in simple terms, can be clearly measured and are defined in the same way as both descriptions of the port nautical chain. The used indicators and the goal value from real world observations are provided in Table 2.

The indicators of Table 2 have been chosen based on the availability criteria of the real world observations and ability to construct mirror indicators on these real world data. If a parameter set is found such that the simulation yields the same key performance indicators (KPI) as observed at the Port of Rotterdam, the calibration is successful.

Apart from these performance indicators there are also some statistics that are directly copied from the observation data. An example of this is the number of visits to a certain berth location and the duration of a stay at that berth. The model will automatically reproduce these numbers accurately (in a long enough simulation), since the distributions are obtained directly from the data.

[4] Automated Identication System – Obliged safety system for all nautical vessels.

Table 2: Indicators with reference value and source on which the model will be calibrated.

Indicator	Agent type	Reference value	Source
Incoming nautical chain duration	Ship, pilot	93 minutes	Kaljouw, 2019
Outgoing nautical chain duration	Ship, pilot	77 minutes	Kaljouw, 2019
Duration of tugging incoming ship	Tugboat	44 minutes	Kaljouw, 2019
Duration of tugging outgoing ship	Tugboat	25 minutes	Kaljouw, 2019
Time between VHF contact for departure and start of departure journey	Ship	16 minutes	Port call data 2016 & 2017

A.2 Mapping the effect of input parameters

To structure and explain the calibration, first the relations between the parameters and calibration indicators have been mapped. The goal of this mapping is to group input parameters such that each group has a collective impact on the calibration indicators. There is however an indirect relation between each group of input parameters and the different calibration indicators. To bridge the gap between the input parameters and the calibration indicators a layer of ‘dependent variables’ has been added to the mapping. For each group a general dependent variable has been defined, see Table 3. Subsequently for each dependent variable the relation to the calibration indicators has been mapped in Table 4.

Pilot service level and Tugboat service level are two of the dependent variables. Service level is an abstract term, but in the calibration and sensitivity analysis the service level is related to the waiting time for services. The lower the waiting time for a service job the higher the service level. As the general mapping of relations between indicator and input requires no hard definition of the dependent variables and is only intended to give insight in the relations.

Another dependent variable is the distribution of visited terminals. This one is directly obtained from observation data. When sufficiently many port calls are made in a run the distribution of visited terminals will automatically be reproduced.

The resting locations of the pilots are not considered to be relevant input parameters here, whereas the resting locations of the tugboats are seen as relevant input parameters. The reason for this is the interest from the Port of Rotterdam in scenario’s with varying numbers and/or locations of tugboat resting locations.

Note that the overview below gives insight in the relations between indicators, parameters and variables in the simulation model. These relations differ from practice because of limitations in the simulation model. For example, in the model the maximum speed of individual tugboats does not influence the duration of the process of tugging incoming or outgoing ships, because in the model the speed of the ship determines the speed of this process. In general, the individual maximum speed of tugboats is higher than the speed of a ship being tugged or pushed by a tugboat.

Table 3: Grouping of relevant input per dependent variable.

Dependent variable	Relevant input parameters and methods
Ship sailing speed at sea Ship sailing speed in port	Ship speed at sea Ship speed in port
Pilot service level	Pilot speed at sea Pilot speed in port **Number of pilots** Method of scheduling*
Tugboat service level	Tugboat speed in port **Number of tugboats** Resting locations of tugboats* Method of scheduling*
Distribution of visited terminals	*None – indicator determined by fixed input data from real world observations. Simulation with large enough sample of port calls will reproduce this distribution.*

*** *nonlinear response to changes and therefore not part of illustration below.*
*[**bold**]* *number of service agents are highly significant input parameter for service levels.*

Table 4: Mapping of relation between calibration indicator and dependent variables.

Indicator	Dependent variable for indicator in simulation
Incoming nautical chain duration	Ship sailing speed Pilot service level Tugboat service level Distribution of visited terminals
Outgoing nautical chain duration	Ship sailing speed Pilot service level Tugboat service level Distribution of visited terminals
Duration of tugging incoming ship	Ship sailing speed Distribution of visited terminals
Duration of tugging outgoing ship	Ship sailing speed Distribution of visited terminals
Time between VHF contact for departure and start of departure journey	Pilot service level Tugboat service level

Figure 27 shows the response of the model to increasing or decreasing the input values of parameters, the following responses are identified:

– Positive response – higher value for the input parameter in theory[5] implies a higher value for dependent variable

[5] Stochasticity in simulation can cause erratic response which differs from the expected positive or negative response. Main difference is in different data samples for indicator calculation in different simulations.

- Negative response – higher value for the dependent variable in theory[6] implies a lower value for the calibration indicator.
- Nonlinear response – the response is not linear and therefore cannot be identified as positive or negative.
- Direct relation – The dependent variable and the input values are directly related and will have the same values.

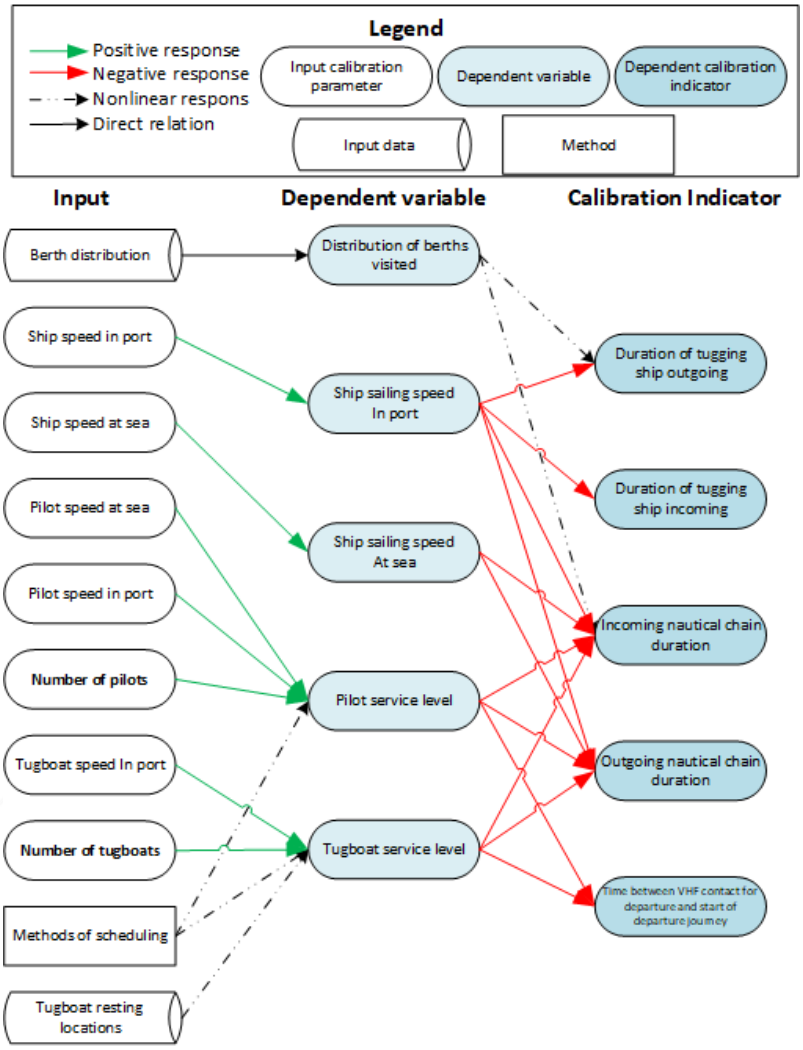

Figure 27: Relation of input via dependent variables to calibration indicators. Number of pilots and tugboats in bold of the significance of these parameters.

A.3 Calibration method - combination of grid search and manual tweaking

Knowing the relation between the input parameters and the performance indicator gives insight in how to change input parameters to make the indicators get closer to the calibration goal. However manual adjustment, running a simulation and

[6] Idem.

analyzing the result is time consuming job. There an automatic grid search algorithm has been implemented to run the simulation for a series of parameter settings, where the parameter values get assigned based upon a predefined grid.

A.3.1 Search for the minimal sum of squared errors

As the goal of the calibration is to fit several simulation KPIs to real world KPIs a fit on multiple values is required. To score results of simulation runs with different parameter settings one uniform scoring value has been determined by summing the squares of the absolute error per KPI, similar to the technique of using the mean squared error (MSE). This sum of all squared errors indicates the quality of simulation results for the parameter set. The lowest sum indicates the parameter set which produced the results closest to the observed performance indicators.

A.3.2 Manual tweaking

When a decent parameter set has been found using the grid search method, some manual adaptions can be performed to improve the parameters further. The method finds a parameter set such that the simulation with these parameters reproduces the performance indicators as accurately as possible. However, one or more KPIs may still show large deviations. Knowing the relations between the input parameter and the indicators (see Figure 1) may help when manual changes to the parameter set are made.

Running a new simulation after a parameter adaption gives a new result which can be compared to the previous simulation results on individual KPI level. This has been executed until no significant improvement could be found for the parameter set. The results can be found in the next chapter.

B Calibration results

This attachment describes the results from the calibration process. Next to numerical results obtained in the calibration process also some remarkable findings are reported in the before last paragraph. As calibrating the model was a process where many aspects of the model have been used and discovered. The most interesting lessons learned in this process are described in the last paragraph. First the main results of the calibration are provided in the paragraphs below.

B.1 Reliability of calibration results

B.1.1. Stochastic distributions

The simulation uses a variety of stochastic distributions. This causes uncertainty in the calibration indicators resulting from a simulation run. These calibration indicators are averages of observed state durations in the simulation and the uncertainty decreases as the sample size of number of observations is increased. However due to computation time limitations the sample size is also limited. To provide insight in the level of uncertainty the calibration and sensitivity analysis have been executed on six different random seeds, to exclude incidental stochastic influence on results and give more certainty to the conclusions from these results.

B.1.2 Simulation starts with an empty port

The simulation model currently starts all simulation with an empty port. The port being empty means also that the first simulated ships never have to wait for a berth or services, this influences the averages shown in calibration indicators.

Ideally the first so many days of registration should be ignored and statistics can then be determined on a system that achieved a balanced state. This has not been done for the SWARMPORT simulation model, because of several reasons. The main reason is that the computation time of the model is quite heavy, meaning that simulating extra days at the start that could be ignored is costly. Another reason is that the number of days before the simulation model gets into a balance is not known and also varies for the given input. It might even be that the system will never be in balance and for example will only produce growing queues.

Due to complex agent interactions and agent states it is not easy to adapt the model to start with already ships inside the port. This in combination with the reasons above lead the choice to compare sensitivity analysis results and calibrate the model including the empty start.

B.1.3 Only completed processes in simulation statistics

The calibration indicators are determined from logfiles of simulated vessels containing duration of the different nautical chain processes and waiting time durations for these processes. In the statistics only process durations of finished processes are taken into account, since for these processes a start time and end time have been logged. This means that for all ships that are in a certain process, for example waiting for a service, when the end of a simulation is reached, no end time is logged and they are not considered in the analysis.

B.2 Determining tick size for calibration

An important setting in the simulation model is the time resolution of the simulation. This parameter is a tradeoff between computation time and exactness of the simulated results. The larger the tick size the larger the rounding errors that will be made in process durations in the simulation, but the higher the simulation speed and the lower the computation time. In the developed implementation of the model and the goal it pursues a tick size of 6 minutes has been chosen. This choice provides a reasonable computation speed and a decent calibration result as described in the previous chapter. This can be seen in Figure 28 and Figure 29. It is important to note that the calibrated set of input parameters is determined for the tick size of 6 minutes. Main reason for the choice is the limited computation time.

In general if tick size if reduced by factor 2, twice as many ticks would be required to simulate the same period of time and the expectation would be that simulation duration would also increase by factor 2. Figure 29 shows that this deduction holds for going from a tick size of 4 to a tick size of 2 minutes per tick, as the simulation duration goes from around 1500 seconds to around 3000 seconds. However, from 8 minutes per tick to 4 minutes per tick the simulation increase does not double. This can be explained that simulation processes that occur outside of the tick events, such as the simulation manager orchestrating the ticks, take up a greater ratio of available processing power. The tick size does not reduce the computation effort of these processes and therefore the processes outside of the tick events become the bottle neck in the simulation speed.

Analysis on varying the tick size shows that the manner of discretization of the processes and discretization of time interact in the simulation model. The discretization of processes of the nautical chain in agent statuses and of sailing routes in node-based network and the parallel simulation in discrete time intervals leads to this interaction. This means that the size of the time interval, the tick size, impacts the calibration and a therefore a fixed value has been chosen that balances computation time and calibration result.

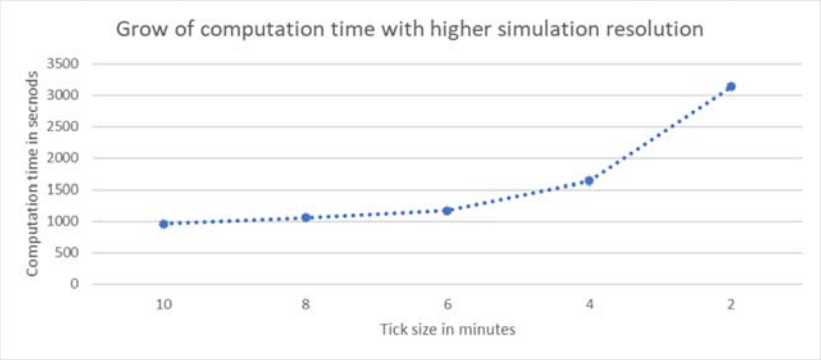

Figure 28: Computation time required for simulating 14 days for increasing resolution, reducing tick interval.

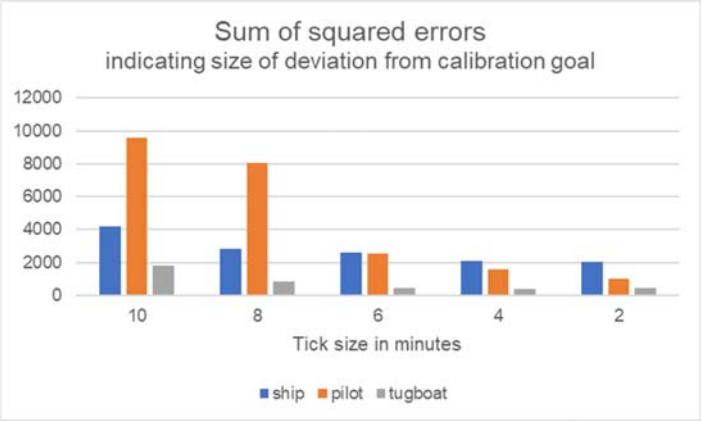

Figure 29: Sum of squared errors for increasing simulation resolution, reducing tick interval. This result has been attained with a sub-optimal set of calibration parameters – before the calibration process was finished – thus a different set of parameters then the final calibration parameters.

B.3 Calibration result

The calibration resulted in a simulation result where all state durations approach the goal values within 15 minutes. The goal values are derived from the reference scenario as described in section 0. Note that the some of the goal values are from this reference are split into part, for example the duration of the ship incoming process has been split in two parts, first the part with the pilot and second the part with pilot and tugboats.

The worst indicating value still deviates 55% from the target value, this is for the tugging outgoing process. With the used parameter set this can no longer be improved, as improving on this indicator would do damage on other calibration indicators. Note that 55% still is relatively high deviation. The simulation model contains an approximation of various real world processes. Each approximation, as the word says, lacks certain real world dependencies or complexity and these approximations could lead to mismatch between model result and real world parameters.

The calibration factors resulting from the calibration process can be found in Table 5 and the effect of these calibration parameters is visualized in Figure 29. This figure shows the value of calibration indicators before calibration – i.e. the values assigned to the variables based on estimates of their true values by the modelling experts, the goal value and the resulting indicator values at the end of the calibration process.

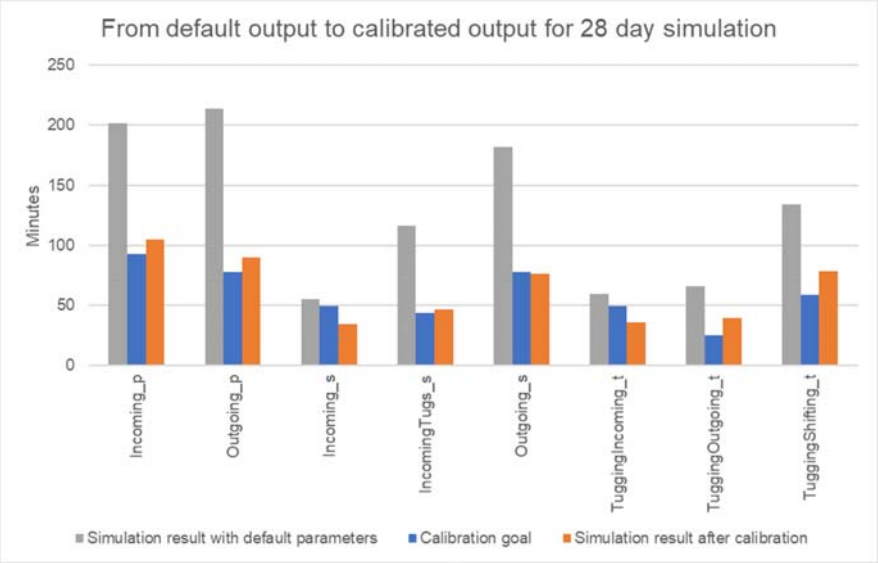

Figure 30: Calibration result of 28 day simulation presented, calibration goal value and simulation result before calibration. Simulations with the same random seed for the stochastic variables.

B.3.1 Numerical results of calibration

The calibrated simulation model shows a significant improvement in the indicator values. However there are still some minor deviations between the goal value and the indicator value. These deviations cannot all be reduced simultaneously by changing the current calibration parameter set. For example the duration of process of 'Pilot Incoming' is higher than the goal value and the duration of the process 'Ship Incoming' is lower than the goal value. Increasing the ship speed modification factor would reduce the 'Pilot Incoming' duration, but also decrease the 'Ship Incoming' duration. Making the 'Pilot Incoming' indicator approach the goal but 'Ship Incoming indicator' deviate further from the goal value. In other words the resulting set of calibration parameters found a result which minimizes deviations from the goal values, but the set of parameters is not extensive enough to reduce all deviations to zero. To achieve this additional calibration factors or even alterations in model choices are required.

Table 5: Calibration factors before and after calibration.

Calibration parameter	**Initial Value**	**Calibration result**
'modification_factor_ship_speed_at_sea_kmh'	1.0	3.0
'modification_factor_ship_speed_in_port_kmh'	1.0	2.75
'modification_factor_speed_at_sea_tug_kmh'	1.0	1.0
'modification_factor_speed_in_port_tug_kmh'	1.0	2.0
'modification_factor_pilot_speed_at_sea_kmh'	1.0	1.0
'modification_factor_pilot_speed_at_land_kmh'	1.0	2.0
Result on calibration indicators	*See Table 6.*	

Table 6: Value and error of calibration indicators before calibration.

Indicator	Agent type	Reference value [minutes]	Error before calibration [minutes / %]		Error after calibration [minutes / %]	
Incoming nautical chain duration	Pilot	93.0	108.3	116%	11.7	13%
Outgoing nautical chain duration	Pilot	77.4	135.5	175%	12.1	16%
Incoming nautical chain duration (1/2)	Ship	49.2	5.9	12%	-15.1	-31%
Incoming nautical chain duration (2/2)	Ship	43.8	72.6	166%	2.8	6%
Outgoing nautical chain duration	Ship	77.4	104.8	135%	-1.1	-1%
Duration of tugging incoming ship	Tugboat	49.2	10.1	21%	-13.6	-28%
Duration of tugging outgoing ship	Tugboat	25.2	40.7	162%	13.7	55%
Duration of tugging shifting ship	Tugboat	58.2	75.5	130%	20.5	35%
Sum of Squared errors				*53837*		*1313*

B.4 Validation of calibration results with Port of Rotterdam

The assumptions for this calibration and the resulting calibrated model are discussed with the Port of Rotterdam. In this validation the calibration results were presented to port nautical chain experts to show and discuss current model outcomes and model shortcomings. The number of pilots and tugboats in the simulation were compared with the reality.

B.4.1 Number of pilots in simulation is valid

In this discussion the Port of Rotterdam shared that as a rule of thumb the pilots ensure that the pilot staff contains five times more pilots as they would require on an average day. In other words if they in general need 10 pilots to be on duty at the same time, they want to have 50 pilots in their staff. This number is to ensure they can provide services 24 hours per day, 7 days per week and it includes overcapacity required to accompany for leave days and sickness of employees.

In the calibrated simulation model there is an active number of 10 pilots. However, these simulated pilots work 24 hours per day and do not have leave days. To validate with practice in the port we compared this with current number of pilots. Currently the pilot organization works with 30 to 35 pilots a day. Assuming a pilot shift is around 8 hours day, means that with the 24 hours a day the 10 simulated pilots can do the work of 30 active pilots in practice. This is the lower bound in the range 30 to 35 pilots a day.

The simulation model is calibrated on 50 vessels per day which all require a pilot. These 50 of vessels per day was retrieved for the Port Call Data of 2016 and 2017 as supplied by the Port of Rotterdam from the Harbor Master Information System (HAMIS). In practice in 2020 around 80 vessels arrive per day of which around 60 ships a day require a pilot. This means the calibrated model has a relatively low ship arrival rate, compared to the 2020 data. Also the number of active pilots is in the lower bounds of current practice. Assuming the ratio of pilots to vessels per day is constant we conclude that having 10 pilots in the simulation model with 50 vessels arriving per day. Where in current practice there are 12 active pilots to 60 vessels per day.

B.4.2 Tugboats operate not only in the port nautical chain

In current practice around 28 tugboats sail in the Port of Rotterdam. This number is little higher than the 20 tugboats in the simulation model. However of the 28 tugboats in the Port of Rotterdam it is currently not known if they are always fully in operation for the nautical service chain. Where the 20 tugboats in the simulation are just like the pilots 24 hours per day and 7 days per week available for nautical services, where in reality time is required for among others maintenance and bunkering.

For now not enough information is available to conclude 20 tugboats is a valid number, but the order of magnitude is surely correct. Concerning this number of tugboats It is good to note that the simulation model contains only 1 tugboat organization, whereas in practice 2 organizations are active. However, the Port of Rotterdam shared that 1 of the 2 organizations does the majority of the tugging and having only 1 tug organization in the simulation is not completely wrong.
The sensitivity analysis on the number of tugboats in the simulation model sheds some more light on if 20 tugboats is reasonable service capacity.

B.5 Meaning of calibration results

The resulting calibrated parameters show that the sailing speeds in the simulation had to be increased drastically to ensure process durations in the simulation match with real world durations. This paragraph gives some meaning and explanation to this calibration result.

The high modification factor for the sailing speeds in the simulation shows that there are more delays than shortcuts in the simulation. The different elements in the simulation that cause delays in the process are identified in the second paragraph below. The first paragraph below goes into the default sailing speeds which are adjusted with the calibrated modification factors.

B.5.1 Default input values of sailing and travelling speeds

The default parameters for the speeds of vessels and services are derived from observations in the AIS data. These values can therefore be considered as a good representation of real world speeds.

One downside is that these speeds do not represent the lower speed of processes at the start and end of processes. A ship will not sail full speed towards a berth during the mooring process. This is one of many details that cannot be all integrated in a model that make calibration of the model essential to catch the lack of details in

the set of calibration parameters. The model speeds are an approximation of the duration of port nautical chain processes and is to aggregated to contain speeds for all sub speeds in

B.5.2 Elements of delay that have to be compensated

In the simulation model some implemented processes take longer in simulated time than they would take in reality. This causes some delays in the simulated port nautical chain processes that had to be overcome with the calibration parameters. The main sources of delay are:

- Discrete simulation steps;
- Scheduling of service agents.

B.5.3 Discrete simulation steps

The simulation is structured in discrete time steps to ensure the various simulated agents are synchronized. This is required because the agent interactions in the nautical service chain should occur in a synchronized manner.

The discrete steps make that some interaction processes or agent state changes take at least one discrete time step in the simulation. Where in reality this might happen in a split second. As the nautical service chain require multiple agent interactions and state changes there are several points of delay in the simulation. This is the main reason the calibration results in high modification factors for speeds.

B.5.4 Scheduling of service agents

The implemented logic for scheduling of services to vessels is a first-come first-serve method. This results in sub-optimal service agent assignments and most likely a worse assignment then real world planners can make. This causes additional waiting times and delays in the simulation. The greediness of the assignment can cause grid lock situations as the assignment of tugboats and the assignment of pilots are not aligned. This can cause all pilots to be assigned to incoming vessels which are waiting for tugboats while all tugboats are assigned to departing or shifting vessels, leading to grid lock situation. To prevent this latter from occurring the number of ships that can be in the incoming nautical chain at the same time is limited to the number of pilots. The queue for incoming pilotage is therewith joint with the queue for clearance.

B.6 Remarkable findings during the calibration process

In the calibration process many simulation runs were executed. Some of the executed runs or analysis lead to remarkable outcomes. Some outcomes were model shortcomings that have been repaired in the calibration process, but others were remarkable results coming from made assumptions and choices in model implementation or input data.

From these latter the six most interesting have been mentioned below:

1. The set of calibration parameters is limited, which limits the effects on calibration indicators. The limitation arises in that the calibration resulted in a balance between calibration indicators excessing the goal value and other indicators ending below the goal values, where the calibration parameters

can only affect the balanced indicators simultaneously. Changing the calibration parameter leads to a reduced calibration result and disturbs this balance. The reason behind this limitation is the available real world observations for model calibration. More insight and detail of service process could improve model quality.

2. The width of the discrete time intervals in which the simulation runs has a major impact on the performance indicators. The impact is either beneficial for the calibration or otherwise hindersome. As the time intervals round process durations to whole intervals, but also make minimal process durations the size of one time interval.
3. Berth saturation gridlock might occur when shifters wait for each other's berths to become available and meanwhile claim all service capacity. This shows that the model is a limited approximation of the reality and not all aspects of agent logic can be incorporated into the model. A more detailed analysis and implementation of agent's logic and procedures would be beneficial for the model quality, see the section of further research.
4. Gridlock might occur when all tugboats were assigned to outgoing or shifting vessels and all pilots were assigned to incoming vessels, see the previous point on the ways to alleviate the issue
5. Some terminals get more ship visits a year than others. Each terminal has limited number of berths and therefore a limited capacity. Using historical visit distribution and provided berths per terminal leads to long queues for terminals with lots of yearly ship visits but limited berthing space.

C Sensitivity analysis

Next to showing the capabilities of the model to represent reality we also show the reaction of the model to changes in input parameters. To verify that the implemented logic in the model behaves accordingly.

Therefore the following three types of sensitivity analyses have been executed:

1. Impact of time resolution of simulation; size of simulation tick in minutes
2. Impact of number of service agents on service level in the port nautical chain
3. Impact of number of arriving ships on the port performance.

Part of the results of the sensitivity analysis of tick sizes has been placed in chapter 4, as it turned out to be an element which interacted with the calibration, see section B.2. Note that the bandwidth of parameters changes in the scenarios for sensitivity lays outside of what is realistic in current practice. These scenarios have been simulated to show how the model reacts to these scenarios and not to analyze scenarios for practical use.

C.1 Sensitivity of port performance for varying service capacity

This sensitivity analysis has been executed by varying the number of service agents, in more detail the number of pilots and the number of tugboats, in the simulation model and analyzing the impact on the model results. The parameters for number of service agents have been chosen because they have a significant impact on the performance of the port nautical chain, which can be easily identified in the simulation model performance indicators. At first the service level should increase if the number of service agents increases and secondly in parallel with an increased service level ships should clearly spend less time in the port nautical chain process. This response on simulation performance can be directly observed in the simulation performance indicators.

The port nautical chain can be seen as a job-scheduling process, where the vessels are the jobs and the pilots and tugboats are servers. Increasing the capacity of the servers should have a reducing effect on the waiting times and the total nautical chain duration. The total nautical chain duration can be derived from the simulation model performance indicators. The waiting times for the services are registered by the ship agents in the simulation and from these waiting time registration a service level indicator has been calculated.

C.1.1 Calculation of loss time

In the simulation model the performance of the port nautical chain services is being scored with the indicator ‘loss time’. The loss time is defined by the time that ships have to wait for a specific service type: clearance, pilotage or tugging.

In the simulation model, the choice has been made, that the number of vessels that get clearance is limited by the number of pilots in the port. This measure is required to prevent grid lock situations in the agent-based system. This decision has an impact on the loss times for pilots which are recorded in the simulation results.

The actual waiting time for a pilot can no longer directly be observed, because part of the waiting time for a pilot is now combined with the waiting time for clearance. The simulation model also records the waiting time for clearance, however in this registration it is not defined which share is caused by a pilot shortage.

C.1.2 Results of sensitivity analysis on number of service agents

The parameter number of pilots and tugboats have been varied for this sensitivity analysis. The reference scenario, resulting from the calibration as described in the previous chapter, contains 10 pilot agent and 20 tugboat agents. For this sensitivity analysis these parameters have been varied around these reference values to show how the model reacts on parameter changes.

The parameter number of pilots has been varied from 4 pilots up to 16 pilots. and number of tugboats have been verified resulting in the service level changes seen in Figure 31 and change in performance indicators as seen in Figure 32.

The number of tugboats have been varied from 10 up to 40 tugboats resulting in the service level changes in Figure 33 and performance changes in Figure 34.

C.1.3 Sensitivity analysis of number of pilots

Varying the number of pilot agents should have a significant impact on the port performance. As they are an essential element in the port nautical chain and reducing the number of service agents would directly reduce the capacity. This is confirmed by the simulation model with the results in Figure 31. The average turnaround time and total loss times in all the port calls are higher with lower pilot numbers. With a higher number of service agents the system reaches a more stable average turnaround time and number of loss hours. This shows that the main bottle neck of the system is no longer the pilot service capacity.

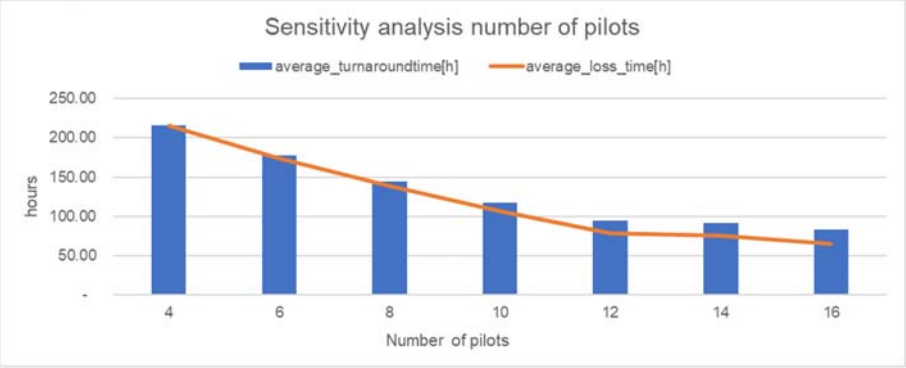

Figure 31: Impact on port performance when varying number of pilot agents, averages of 6 different simulation seeds.

Next to the main port performance also the effect on the loss time per service type and per direction are given in Figure 32.

In this visual the following can be observed:

- The number of vessels that get clearance to start a journey through the port is limited by the number of active pilot agents. This system rule explains the continuous high service levels of the pilot service, as the number of service

requests is limited leads to most request being fulfilled without serious waiting time.

- The loss time caused by the tugboats also varies if the pilot capacity is varied. This can be explained by the fact that the tugboat service is dependent on the speed of service delivered by other services. With few pilots the tugboats are also waiting for the other services, this puts pressure on limited tugboat resources.

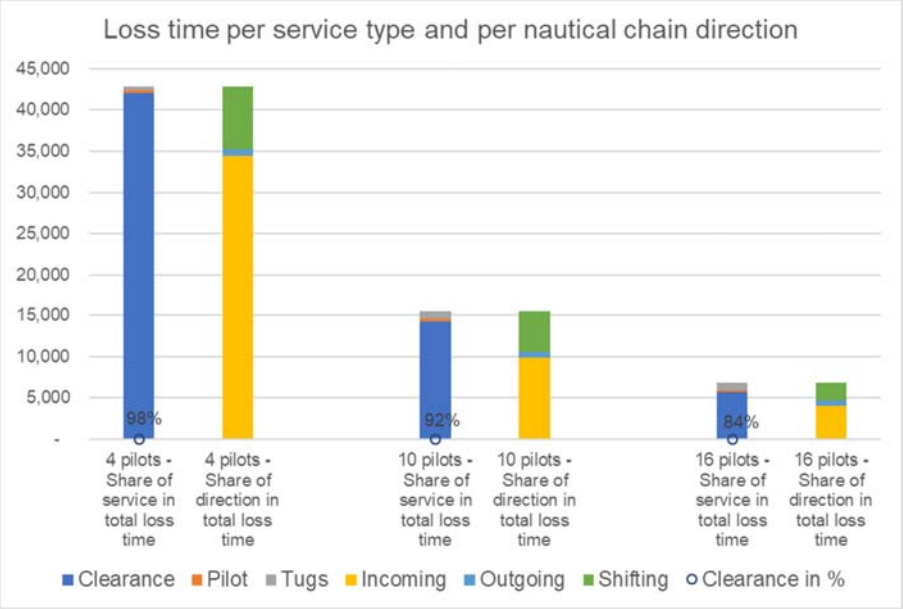

Figure 32: Change in loss time for the different service types and different directions.

C.1.4 *Sensitivity analysis on number of tugboats*

In this analysis the number of tugboats has been varied, similar to the previous analysis where the number of pilots were varied. The effect of this analysis should be similar as the tugboats are also an essential element of the nautical chain process. Reducing their resources and therewith the service capacity should also lead to lower port performance and service levels.

Lower of turnaround time and higher total loss hours can be confirmed in Figure 33. However, the effect is significantly smaller compared to the simulation with a small number of pilots. This can be explained by the fact that the tugboats play a smaller role in the process, the time of tugging is smaller than the time a pilot is aboard the vessels on incoming and departing journeys. Also, differently from the pilots, tugboats do not distinguish nautical chains in ‘incoming’ or ‘outgoing’. Therefore, their resources are more flexible in assignment to service requests.

A remarkable result in this analysis is that the average turnaround time and the average loss time seem to increase for the last increase of tugboats, this is counter intuitive as more service agents should lead to reduction in loss time and turnaround time. A definite explanation for this increase has not been found but possibilities are:

- The analysis has been executed only 6 random seeds and the standard deviation for these averages is quite high, 23.5 on an average of 130.3 hours. So mere stochastic variations could create this effect.
- More tugboats could lead to longer waiting time at pilot services, could explain an increase in loss time, but should not influence total port turnaround time, as number of pilots is the constant.
- More tugboats in the simulation could have the effect that more ships with higher turnaround time, e.g. shifters, manage to get through the port in the 28 day simulation, causing the average turnaround time to increase.

.

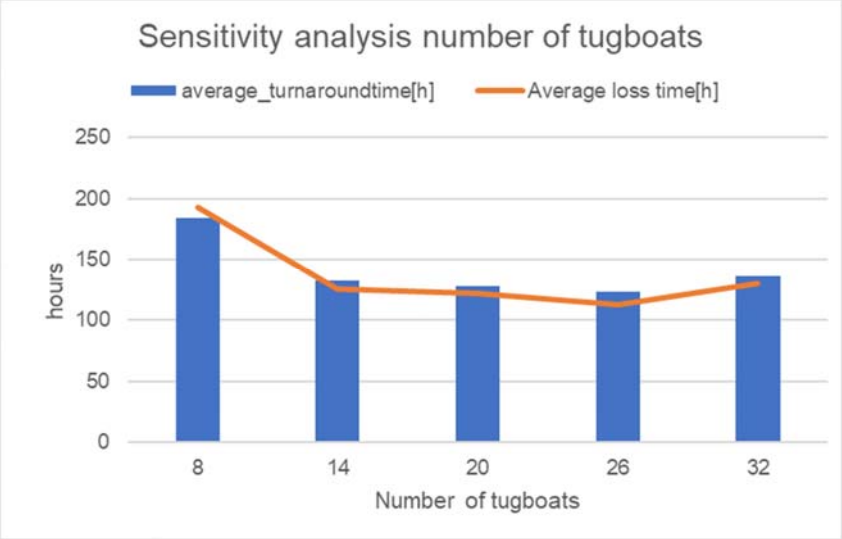

Figure 33: Impact on port performance by varying number of tugboat agents, averages of 6 different simulation seeds.

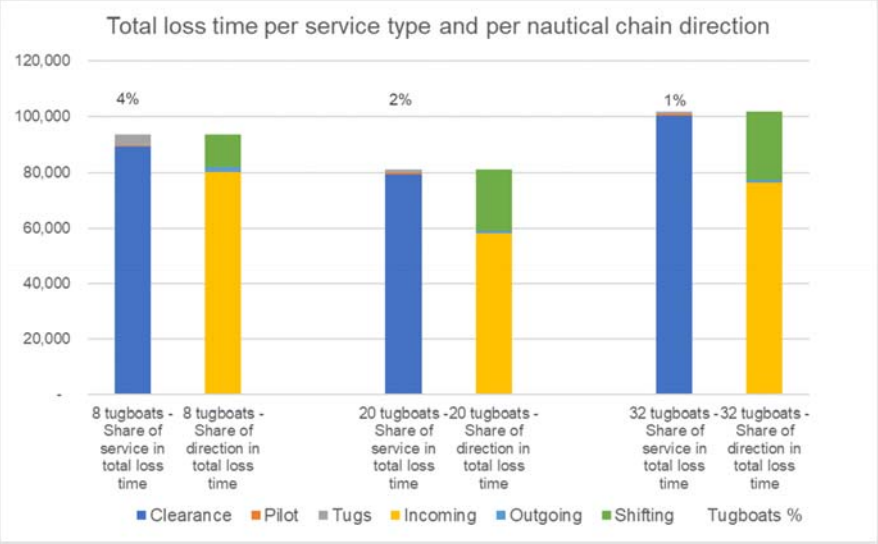

Figure 34: Impact on total waiting times per service type and per direction. Figure shows reduction in share of tugboat waiting hours in total loss time. Increasing total waiting time with more tugboats is a remarkable effect of increasing congestion which could be explained by increased pressure on a bottle neck leading to higher waiting times.

C.2 Sanity check of port performance to arrival rate of vessels

An important factor in the port nautical service chain is the number of ships that make a call at the port. As the more ships visit the port the more service chains have to be executed and more capacity of the nautical services is required. To get insight in how the simulation model reacts to an increasing number of arriving ships an analysis with an increased ship arrival rate has been simulated.

The effects are presented by the impact on average turnaround time and average loss time per port visit, the same as in the sensitivity analyses for the number of pilots and tugboats. The arrival rate has been increased from the default value of 50 ships per day to 75 ships per day. , a massive increase of 50% which should show a lack a capacity in the port nautical chain services. Again simulations have been executed for 6 different random seeds to compare the average performance with 50 vessels per day to 75 vessels per day

Increasing the arrival rate this much should lead to an increase in queue lengths in the simulation model. The simulation model output does not directly show the impact of such a change, as only the waiting times for finished waiting periods are taken into account. Therefore waiting times of ships still waiting in a queue at the end of the simulation do not count in the average waiting time. However indirectly it is possible to see how the increased length of the queues by looking at the ratio of vessels that completed their turnaround and the number of vessels that arrived in the port. The lower the percentage of turnarounds the more vessels that are still enduring the nautical chain process.

Table 7: Comparing simulation results of arrival rate of 50 vessels and 75 vessels per day.

Vessels per day	Average loss time [h]		Standard deviation loss time	% of arrived vessels with completed turnaround
50	115	100%	13.8	49%
75	136	119%	20.8	34%

Table 7 shows the results of the sanity check on arrival rate. An increasing number of arriving ships clearly shows an increasing pressure on the nautical services and berth availability by increasing loss times in the total nautical service chain. Note that the loss time only shows an increase of 19%, which is much lower than the 50% increase in vessels per day. This mainly explained by the ships still waiting in queues in the simulation model at the end of simulation, these have relatively high waiting times which are not yet recorded and therefore not yet accounted in the average loss time. The increase in the standard deviations confirms this as it shows that there is more variation in the waiting times, which are most likely higher waiting times.

D List of calibration parameters

'agent_event_timeout_delay'	0.50
'agent_timeout_iteration_number'	10
'NumberPilots'	10
'NumberTugboats'	20
'ShipArrivalRate1'	50
'MaxShipsInSimulation'	3000
'MinutesPerTick'	6
'SimulationDays'	28
'ship_distance_in_port_factor'	1.50
'ship_distance_at_sea_factor'	1.10
'tug_distance_in_port_factor'	1.50
'tug_distance_at_sea_factor'	1.10
'pilot_distance_at_sea_factor'	1.10
'pilot_distance_in_port_factor'	1.50
'request_hours_before_departure'	1.25
'modification_factor_ship_speed_at_sea_kmh'	3.00
'modification_factor_ship_speed_in_port_kmh'	2.75
'modification_factor_speed_at_sea_tug_kmh'	1.00
'modification_factor_speed_in_port_tug_kmh'	2.00