How to Simulate a MBSE System Model

Using Σ™ and WorldLab™?

Daniel Krob & Antoine Rauzy[1]

Systemic Intelligence Group ©

November 2023

Table of contents

[1] Emails of the authors: daniel.krob@systemic-intelligence.net and antoine.rauzy@systemic-intelligence.net

1. Systemic Intelligence

Systemic Intelligence is a **spin-off** of the industrial chair "Engineering of complex systems" of Ecole Polytechnique. We are specialized in **systems architecting & engineering** and propose **modeling & simulation techniques** to better mastering industrial complexity.

Systemic Intelligence especially disseminated **new methods** in this area for the last 10 years within various international industries, in China, France, Germany & Japan, and developed on this basis an innovative **systemic digital twin technology** dedicated to the optimization of industrial systems. Our current core activity deals with developing and disseminating our new systemic digital twin solution.

Figure 1 – Our ecosystem of industrial customers (our first systemic digital twin customer are squared in red)

Systemic Intelligence is directed by two world leading scientists in the systems engineering domain:

- **Daniel KROB**, chief executive officer of Systemic Intelligence, is a former institute professor in Ecole Polytechnique, the top 1st French engineering university, currently also Distinguished Visiting Professor in Tsinghua University, the top 1st engineering university in China. He is a leading **world expert in system modeling**, recognized as a Fellow of the International Council on Systems Engineering (INCOSE);
- **Antoine RAUZY**, chief scientific & technological officer of Systemic Intelligence, is professor in the engineering university CentraleSupélec in France and in the Norwegian University of Science & Technology in Norway. He is a leading **world expert in system simulation** and especially developed the AltaRica model-based safety technology, currently used worldwide in the industry for supporting safety studies.

Note finally that the **construction of a systemic digital twin** relies on three main scientific pillars that are fully documented (see Figure 2):

1) our **CESAM model-based systems engineering method**, used in the **design phase**,
2) the new **systemic specification language ΣTM** (which shall be pronounced "Sigma"), used in the **beginning of the development phase**,
3) the **WorldLab**™ **platform** that supports the **end of the development phase** and the **use phase**.

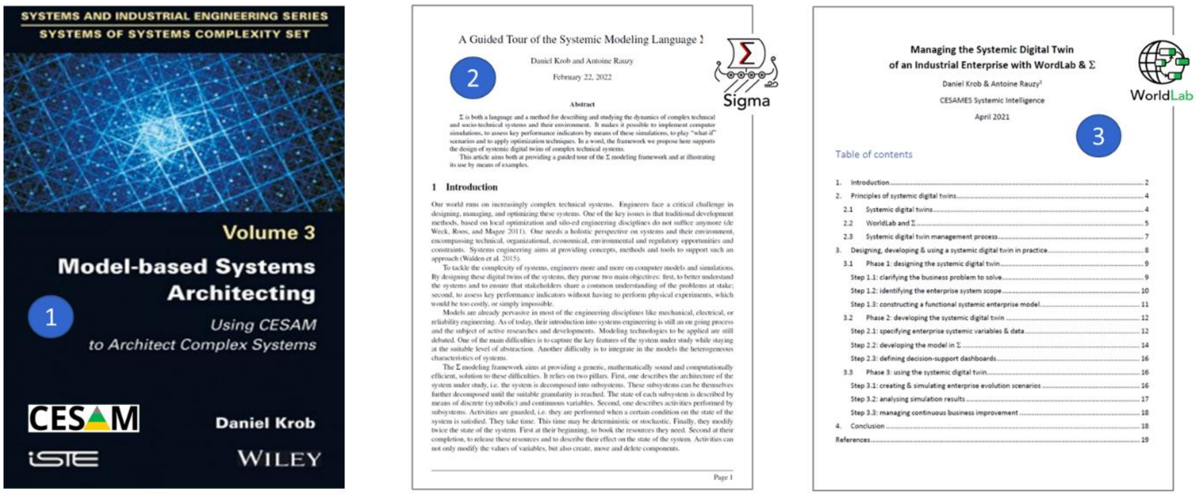


Figure 2 – The scientific pillars of our systemic digital twin technology

2. Introduction to systemic digital twins

2.1 The business scope of a systemic digital twin

Modern industries must **optimize complex interdependent operational ecosystems**, such as their supply chains, their manufacturing systems, their distribution systems, their customer operations, their maintenance systems & policies, etc., taking into consideration **complex economic, political, social, technological, legal & environmental constraints** from a tactical and strategic perspective.

Figure 3 – The business scope of application of our systemic digital twin solution

Optimization of industrial operations typically rely on many different types, first of **strategic industrial decisions** such as:

- What is the optimal global architecture for an industrial system?
- What is the optimal design for a new industrial facility?
- What is the industrial evolution scenario which has the less risks & costs?
- What is the best way to manage an industrial process?
- What is the optimal way to manage an industrial ramp-up?
- What is the optimal industrial maintenance strategy to follow?

but also of many **operational & tactical decisions** such as:

- How to optimize my industrial lead time during operations?
- How to minimize non quality during industrial operations?
- How to determine the root causes of an operational inefficiency?
- How to optimally reconfigure my industrial production?
- How to minimize energy & wastes during industrial operations?
- How to decrease environmental footprint during industrial operations?

Systemic digital twins can in particular be seen as key **decision-aid tools** that can support these types of decisions in complex industrial environments.

2.2 The business scope of a systemic digital twin

The current digital twin market solutions can be characterized by:

- their **scope of application** that can be either *industrial products*, to support their design, or *industrial processes*, to support manufacturing, maintenance & operations,
- their **mode of representation** of a system, that can be either *geometric*, to see where are located the system components, or *behavioural*, to represent what a system is doing.

When crossing together these features, one gets immediately four totally different types of digital twins, as illustrated on Figure 4, that is to say:

- *geometric digital twins of industrial products*, classically called digital mock-ups, which are the most widely disseminated types of digital twins within the industry,
- *geometric digital twins of industrial processes,* that are digital mock-ups of industrial facilities which must integrate the temporal dynamics of the associated industrial processes,
- *functional digital twins of industrial products,* which are either descriptive, leading us to model-based systems engineering tools, or dynamic, supporting multi-physical simulation,
- *functional digital twins of industrial processes,* which is the core business domain of application of systemic digital twins.

Note that the mathematics behind functional digital twins of industrial products & processes are just radically different: on one hand, functional digital twins of industrial products rely – when going to simulation – on numerical resolution of partial differential equations, which lead us to continuous mathematics, when, on the other hand, functional digital twins of industrial processes require discrete event simulation, based on discrete mathematics. Since continuous & discrete mathematics are totally different mathematical paradigms, like water & fire, one can easily understand that systemic digital twins form a specific category of digital twins. Moreover, only 2 % of the current digital twin market solutions are covering this segment, though it is of crucial industrial importance.

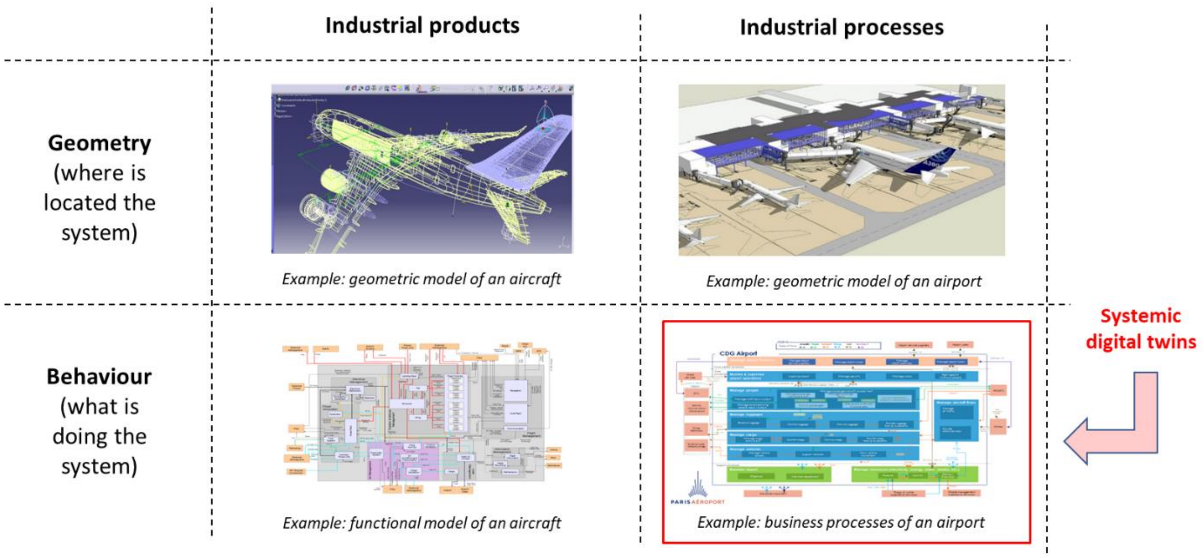


Figure 4 – The functional scope of our systemic digital twin solution

Contrarily to the market (e.g. Ansys, Bosch, Dassault Systèmes, PTC, Siemens, etc.) that focuses either on data-related infrastructure or on geometric representations, we indeed do believe that digital twins for industrial processes must use a **functional point of view:** they shall be able to **model & simulate the behavior, i.e. the business processes, of an industrial system**, starting from operational data and ending by enriching decision dashboards or digital mock-ups, which here puts business models at the core of a digital twin. This is why we took an **enterprise architecture behavioral approach,** which is our key difference with respect to existing digital twin technology for industrial processes.

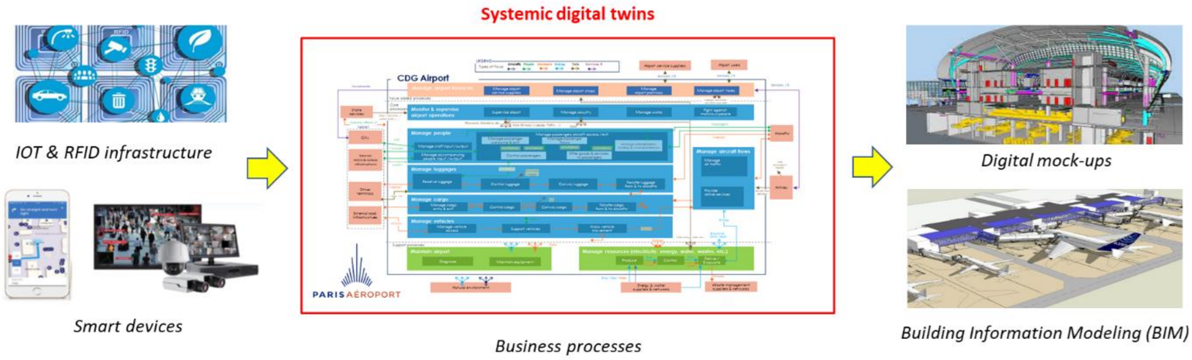


Figure 5 – Our functional digital twin philosophy where business processes are at the core of a digital twin

2.3 The technological scope of a systemic digital twin

To support our vision, we developed the **WorldLab™ patented technology** – built on the **proven infrastructure of the AltaRica safety & reliability analysis tool**, developed by Antoine RAUZY during the last 20 years and industrially used in many industrial sectors – which is a **systemic intelligence workshop** that offers systemic modelling and scenario stochastic simulation & evaluation capabilities.

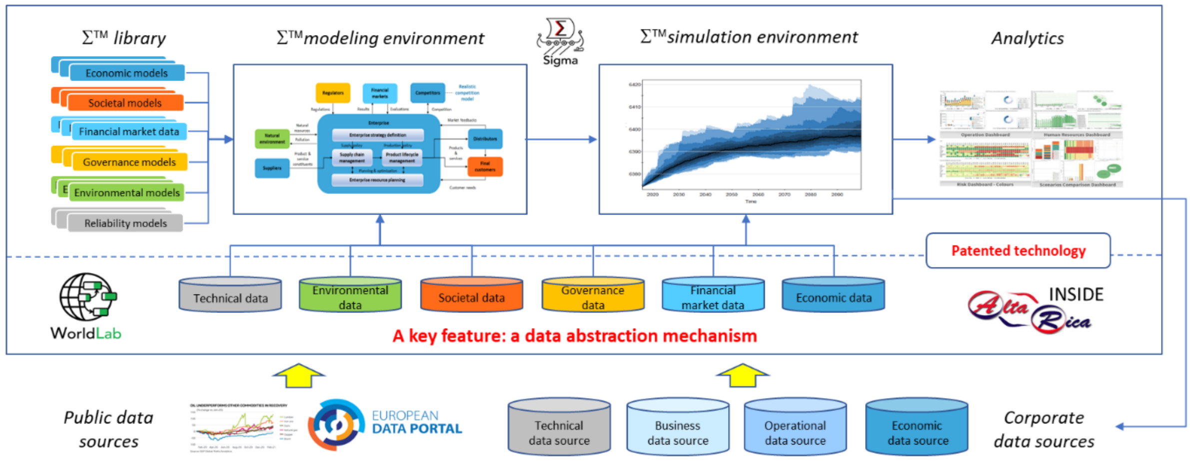
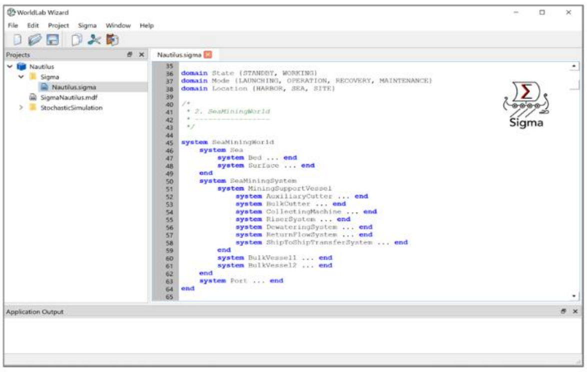


Figure 6 – Overview of the architecture of WorldLab™

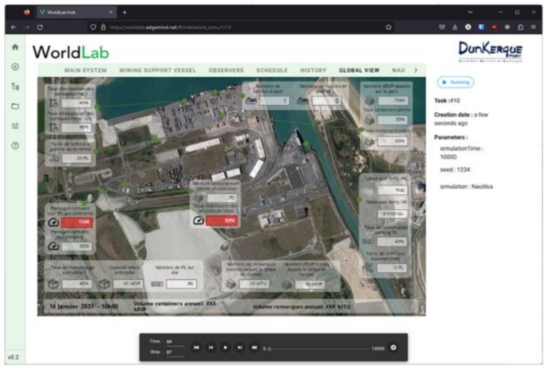
The **WorldLab™ technology** has in particular two sides dedicated to two different types of users, as illustrated on Figure 7:

1) the **WorldLab™ Workshop** is a system modeling & simulation standalone workshop where a **system modeling engineer** can model a given industrial system, using our system specification language Σ™, and prototype the associated systemic digital twin,

2) the **WorldLab™ Hub**, generated by the WorldLab™ Workshop, is the Web interface dedicated to the **business users** where one can simulate a systemic digital twin, evaluate business indicators and compare business scenarios associated with the modeled industrial system.



WorldLab ™ Workshop



WorldLab ™ Hub

Figure 7 – The two faces of WorldLab™

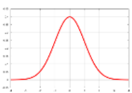
2.4 The key unique features of WorldLab™

The **WorldLab™ systemic digital twin technology** has a number of absolutely unique features that are synthesized here below.

- **Simplicity & Maintainability –** A systemic digital twin is specified in the **object-oriented modeling language Σ™** which is quite simple to use to any person with an algorithmic-design background. This choice allows to **easily maintain the evolution** of a systemic digital twin among time, its evolutive maintenance becoming now totally similar to what is classically done in software engineering.

- **Heterogeneity –** Systemic digital twins can integrate many **heterogeneous features**, such as technical functions, maintenance policies, societal behaviors, geopolitical concerns, financial market evolutions, regulatory strategies or even meteorologic conditions, into a **single unique systemic model**, allowing to analyze an industrial system, taking into account all these various perspectives.

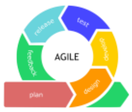
- **Concurrency & Time –** The Σ™ system modeling language especially allows to manage **concurrent industrial activities** and to express explicit durations for **timed transformation activities** of an industrial system, which is currently not offered by the existing similar languages.



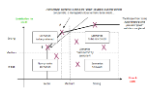
- **Hazards – Hazards** can be effectively captured in a systemic digital twin within our approach: each variable specified in the Σ™ modeling language can be a random variable with a specific probability distribution – either explicit or pragmatic – allowing to capture **random quantities & random delays** and to manage **stochastic simulations** to compute accurate KPIs for a given industrial system.

- **Data Abstraction – Operational data** can be managed – when necessary – through **abstraction mechanisms** that allow to avoid dealing with details when they are not mandatory, while focusing on the most important trends captured by the data. This possibility also allows to gain into execution performance when one needs to deal with complex simulations.

- **Automatic Generation –** A given systemic digital twin can be **automatically generated** from its ΣTM specification which can support **systemic simulation** due to its underlying mathematical foundations & formal semantics (which are however fully hidden to the user who does not need to know them).



- **Quick Development –** The flexible mechanism provided by the ΣTM specification language allows to **quickly develop** in an agile way, typically within a few weeks, a first usable version of a systemic digital twin, based on only some thousands of lines of ΣTM, as soon as the system modeling phase is finished.



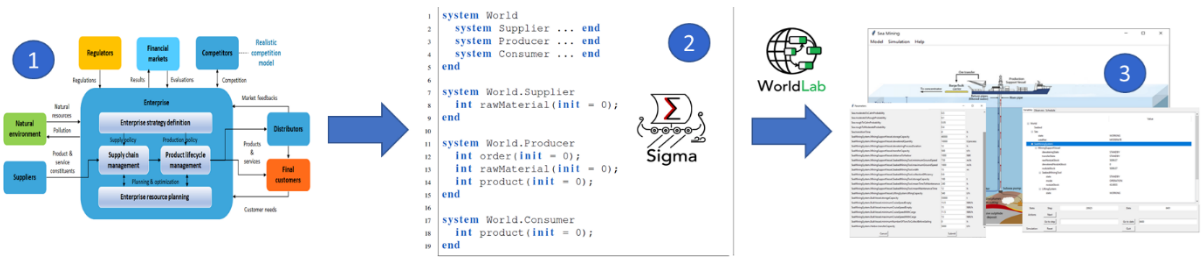
- **Scenario Evaluation & Prioritization –** WorldLabTM platform proposes dedicated features for **evaluating & prioritizing business evolution scenarios**, which allows to achieve **multi-criteria optimization**, e.g. maximizing production and minimizing delays & energy consumption, with respect to a given industrial system.

- **Dashboards & Alerts – Dashboarding and alerting mechanisms** allow to support both **operational & strategic decisions** and also to identify the **deviations** of a given industrial system, when in operations, with respect to its normal trajectory depending on its environment behavior.

- **Methodology –** Last, but not least, a **strong methodological environment**, covering **design & development techniques, environment & world modeling methods and systemic data modeling**, is offered to all modeling users of the WorldLabTM and ΣTM technology.

2.5 Synthesis: systemic digital twins connect MBSE to simulation

As a key already mentioned feature, the **WorldLab™ technology** especially allows to **automatically produce systemic digital twins** of an industrial system from a **MBSE model** through a **specification** designed in our **Σ™ formal modeling language**.



MBSE model of an industrial system

ΣTM model of an industrial system

Systemic digital twin of an Industrial system

Figure 8 – Principle of the development of a systemic digital twin of an industrial system with ΣTM and WorldLab™

The **construction** of a systemic digital twin based on WorldLab™ and Σ™ follows indeed a **standard methodology in 3 steps**, as described here below.

CESAM

- **Step 1 – Designing the systemic digital twin:** the first step for constructing a systemic digital twin for an industrial system is a **model-based systems engineering (MBSE) activity**, based on our CESAM methodology. It consists in clarifying the business problem to solve, identifying the exact business & technical scope to be covered by a systemic digital twin within an industrial system and constructing a functional model of the target industrial system. Its deliverable is a **MBSE model** of the considered industrial system.

- **Step 2 – Developing the systemic digital twin:** the second step consists then in **generating a systemic digital twin** of an industrial system, based on a Σ™ model, obtained from the MBSE model constructing in the first step. One has here to specify the relevant business variables & data, to develop the Σ™ model of the industrial system of interest and to specify the graphic interfaces for the end-users with focus on the decision-support dashboards. The deliverable is a **systemic digital twin** for the considered industrial system.

- **Step 3 – Using the systemic digital twin:** the third & last step of our process focuses finally on the **use of the systemic digital twin**. It consists in creating & simulating evolution scenarios for the industrial system of interest and analyzing the results provided by the simulations in order to manage continuous business improvements and to prove the business value of the systemic digital twin. The deliverable is now a **set of key performance indicators** for different evolution scenarios together with **business recommendations** for the considered industrial system.

2.6 The case study that we shall now follow: a port transformation

The case study that we shall follow in the next sections in order to illustrate our approach is **Dunkirk's port** which is currently the **very first French port for the import of coal**.

Figure 9 – Dunkirk's port

Due to **environmental regulations**, the **old coal traffic is being replaced by a new container traffic** (see Figure 10), which has a **huge impact** on the port infrastructures since coal and containers require totally different logistics, namely **train & truck logistics**. There is thus a strong need to **identify & secure the investments** that have to be done by the port in order to adapt it to the new traffic and to manage the decreasing of the old traffic.

Coal traffic *Container traffic*

Figure 10 – The transformation of Dunkirk's port

3. How to initiate a systemic digital twin?

In order to move towards a systemic digital twin for an industrial system, the very first phase is an initiation phase that intends understanding the relevant scope of interest. It concretely consists in **fully understanding** the **problem(s) to be solved** and the **relevant scopes** from **1.1) strategic**, **1.2) business & technical** and **1.3) data** perspectives, as illustrated in Figure 11.

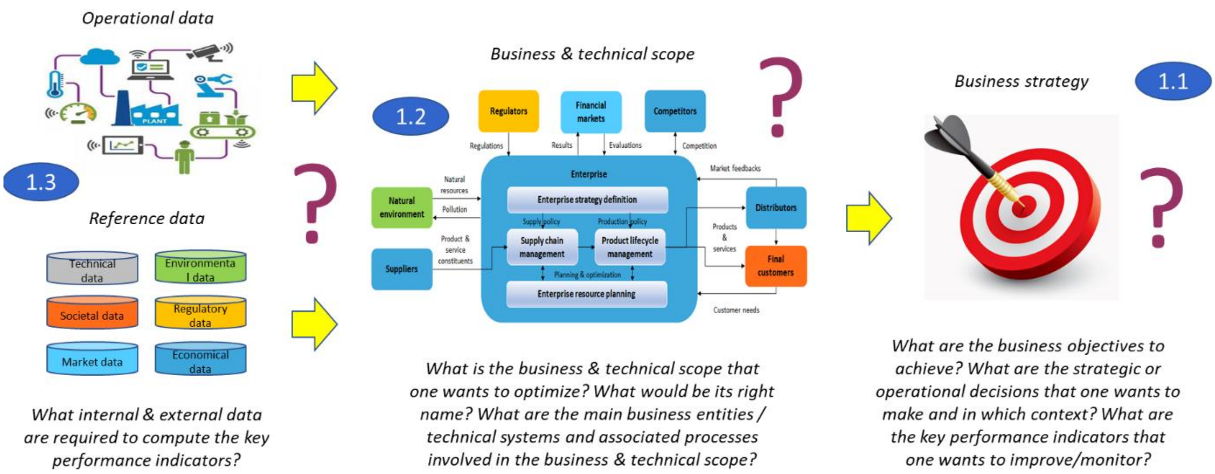


Figure 11 – Overview of the initiation phase

3.1 Activity 1.1: eliciting the business strategy

In order to construct a systemic digital twin, the very first activity consists in eliciting the business strategy which supports its construction by finding the answers to the following questions:

- What are the business objectives to achieve?
- What are the strategic or operational decisions that one wants to make and in which context?
- What are the key performance indicators (KPIs) that one wants to improve/monitor?

The above table elicits for instance the **business strategy** of Dunkirk's port with respect to its **coal-to-container transformation**.

Key questions to answer	Business strategy of Dunkirk's port
What are the business objectives to achieve?	The main business objective of Dunkirk's port is to secure its investments, as required from switching from a coal-dominant to a container-dominant traffic, by verifying that the "to-be" port infrastructures will be resilient against the strong increase in container volumes which is expected from now to 2035.
What are the strategic or operational decisions that one wants to make and in which context?	Since the container traffic shall be managed by trucks that shall leave the port either by road, or by train, the key decisions that Dunkirk's port has to take are of two kinds: 1) How to transform & optimize the existing infrastructures of the container terminal in order to face the expected increasing of the container traffic? 2) How to size a new rail-road terminal that shall manage a part of the traffic induced by the future development of the container traffic?

What are the key performance indicators (KPIs) that one wants to improve/monitor?	• KPI #1: amount of new port infrastructures (cranes, gates, reach stackers) which are required by the transformation of the port • KPI #2: customer quality of service (queuing time, volume of managed containers) • KPI #3: level of pollution induced by the new port infrastructures

Table 1 – Elicitation of the business strategy of Dunkirk's port

3.2 Activity 1.2: modeling the industrial system of interest

The second activity of the initiation phase of the construction of a systemic digital twin consists in modeling the industrial system of interest, which can be achieved through a series of five inter-related **modeling steps**, consisting in starting by 1) a preliminary draft and 2) a geometric scoping and eliciting 3) the system breakdown, 4) its functional interactions and 5) its use cases (see Figure 12).

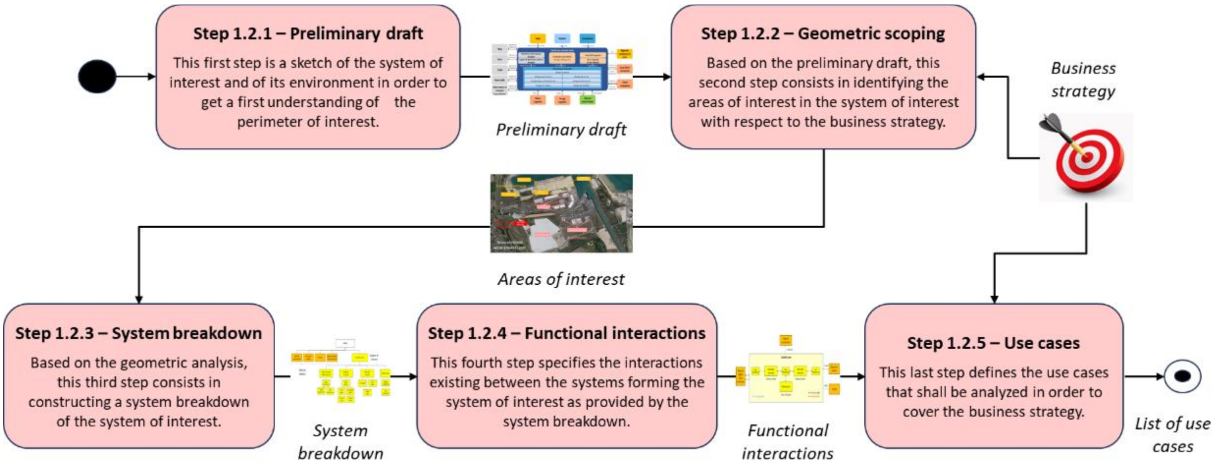


Figure 12 – Overview of the modelling activities of the industrial system of interest

3.2.1 Step 1.2.1: preliminary draft

Constructing a **preliminary draft system model** of the **business & technical scope** of a systemic digital twin is a good practice that helps understanding the system of interest. To this aim, one shall be able to synthesize in a single draft system model the answers to the following **key questions** relatively to the system of interest:

1) what are its **key customers**?
2) what are its **key suppliers**?
3) what are its **key resources**?
4) what are the **key constraints** that it has to manage and from where are they coming?
5) what are its **key business processes** with respect to its business objectives?

The synthesis of this analysis is a draft environment diagram, as illustrated on Figure 13

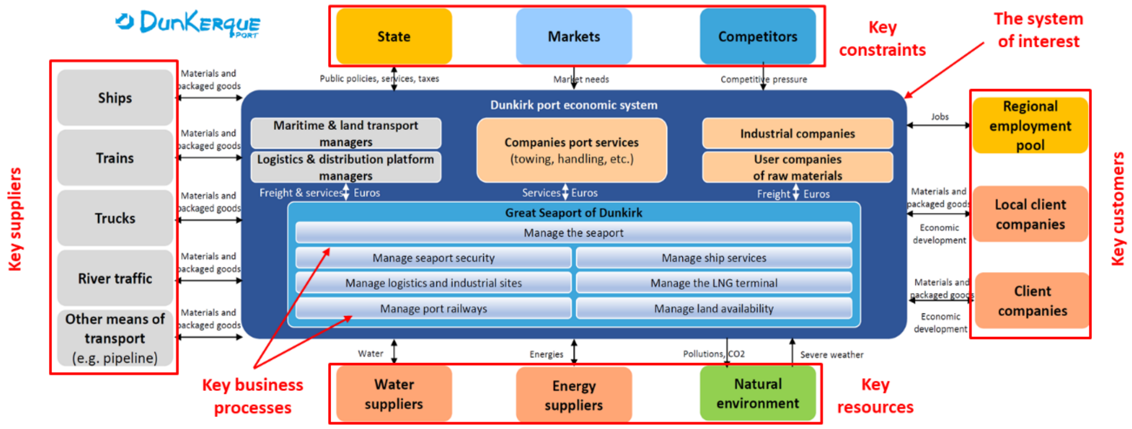


Figure 13 – Preliminary draft of Dunkirk's port industrial environment

3.2.2 Step 1.2.2: geometric scoping

The **geometric scoping analysis** aims then at **identifying the areas of interest**, depending on the business strategy that was previously elicited, within the system of interest that was sketched in the previous step through a preliminary draft system model.

The result of such a geometric scoping analysis is illustrated on the Dunkirk's port case in Figure 14. One can see that the geometric scoping allows here to identify three main areas of interest within Dunkirk's port which correspond to the areas impacted by its transformation:

- A *container terminal*, where the containers are transshipped from & to the ships coming in the port, stored and managed by trucks,
- A number of *warehouses* where the goods contained in the containers can be stored,
- A *rail-road terminal* that has to be constructed in order to manage a part of the growth of the container traffic.

Figure 14 – Geometric scoping of Dunkirk's port industrial environment

3.2.3 Step 1.2.3: system breakdown

The next step of the modeling of the system of interest consists in deriving from the geometrical analysis the **hierarchy of the systems** that are involved in the environment of the system of interest, or in other terms in **defining the system breakdown** of the perimeter of interest using a system breakdown structure diagram.

In the context of Dunkirk's port transformation, this system breakdown analysis leads to the system breakdown structure illustrated in Figure 15. One can see there the complete breakdown of the perimeter of interest – called here "World" – that synthesis all the systems involved or impacted by the transformation of the port. This system breakdown splits in two parts:

- Dunkirk's port which can broken down in four subsystems:
  - The internal road infrastructure of Dunkirk's port used by the trucks that are transporting by road the containers managed by the port,
  - The container terminal where the containers managed by the port are coming & leaving by sea through a specific cargo logistic, stored in a container storage park and coming & leaving by road through trucks that enter in the container terminal by the terminal access,
  - A number of existing & new expected warehouses, where trucks are bringing & taking containers in order their goods to be stored there,
  - A new expected rail-road terminal consisting in train & truck logistic facilities and in a stock of reach-stackers which are specific machines used for moving containers between trains and trucks.
- Three external systems, that is to say:
  - The natural environment that will be especially impacted by the pollution produced by the increasing of the truck traffic induced by the growth of the container traffic,
  - The sea from / to where are coming / leaving the ships that transport the containers that are managed by Dunkirk's port,
  - Two external road & rail infrastructures, respectively used by the trucks and railways that are involved in the management of the container traffic of Dunkirk's port.

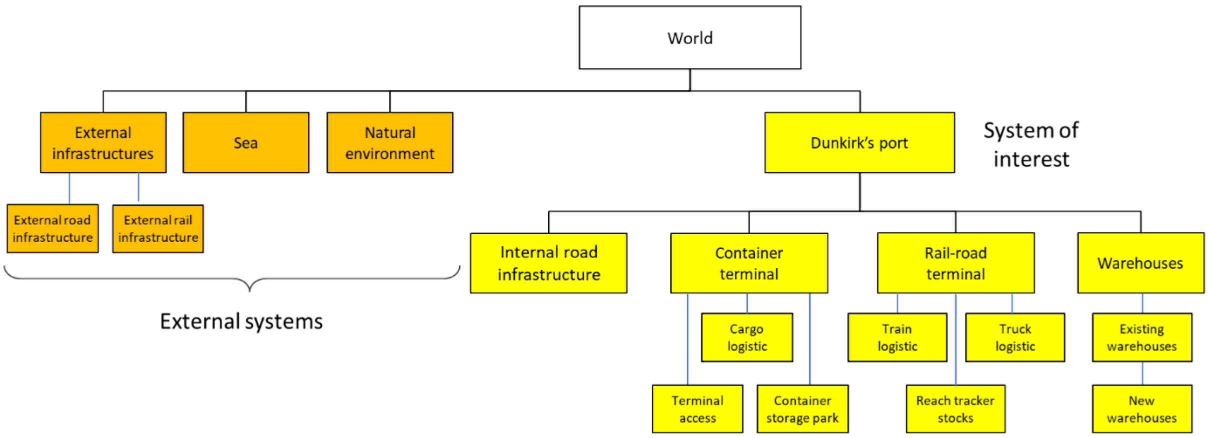


Figure 15 – System breakdown structure of Dunkirk's port industrial environment

3.2.4 Step 1.2.4: functional interactions

The fourth step of the modeling of the system of interest now deals with the **functional interactions analysis** which aims at identifying the structuring **interactions & flows**, that do exist within the various systems forming the environment of the system of interest, and the core **activities**, that are performed by these systems. It especially results in a **functional interaction diagram**, that synthesizes all the interactions existing between the key subsystems of the system of interest and its external systems, and also highlights the key activities achieved by these subsystems.

In the context of Dunkirk's port transformation, this new functional interactions analysis leads to the functional interaction diagram illustrated in Figure 16. This new diagram highlights the main functional chain involved in Dunkirk's case, that is to say the fact that the container traffic is triggered by the sea from / to where are coming / going the ships that are transporting the containers and handled by the port and by external road & railway infrastructures. It especially shows that the port internal container two-way traffic is managed through a container terminal, connected to the sea and to the ship traffic, where containers are therefore received and sent, and an internal road infrastructure for trucks, which connects the container terminal to the external road infrastructure, but also to a rail-road terminal, itself connected to the external rail infrastructure, where trains are received and sent, and to internal warehouses that can be seen as temporary buffers where the goods contained in the containers are stored & destored among time. Note finally that we put there a specific stress on the pollution flow that goes from the port to the natural environment.

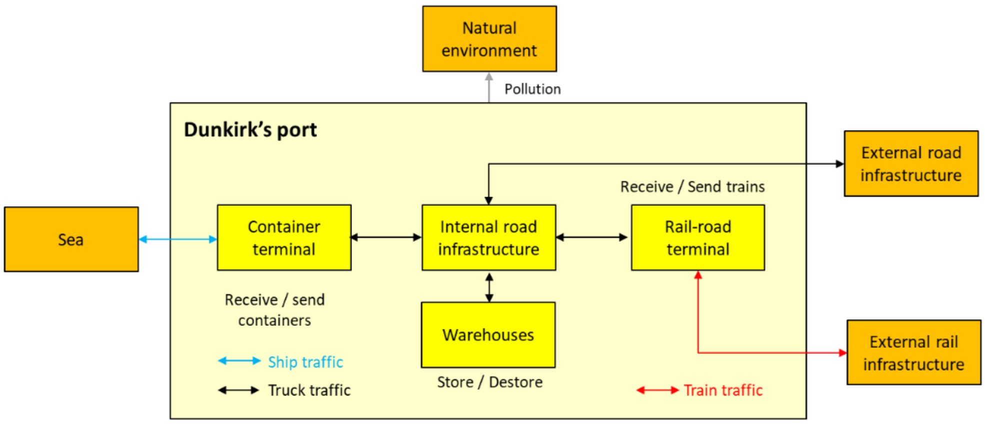


Figure 16 – Functional interactions involved in Dunkirk's port industrial environment

3.2.5 Step 1.2.5: use cases

The last modeling step consists in **formalizing the business strategy** in terms of **use cases** of the system of interest that shall be modeled, simulated & analyzed with a systemic digital twin in a next phase.

In the context of Dunkirk's port transformation, these use cases are for instance the following ones:

Dunkirk's container terminal

- **Use case 1** – *Sizing of the unloading cranes fleet within the container terminal* – The increasing of the container traffic will require to add new unloading cranes to the port, which is a very expensive investment that has to be finely planed. How much new cranes are therefore required and when shall they be put in service?
- **Use case 2** – *Optimization of the truck access to the container terminal* – The increasing of the container traffic will increase the truck traffic on the

port. How shall one thus reorganize the truck traffic management in order to optimize the in/out access of trucks to the container terminal and to provide an optimal quality of service to the trucks?

Dunkirk's rail-road terminal

- **Use case 3** – *Optimization of rail-road terminal infrastructure* – The increasing of the container traffic will require a new rail-road terminal within the port. How shall one therefore organize optimally this new terminal at the interface between trucks and railways in terms of specific loading/unloading machines?

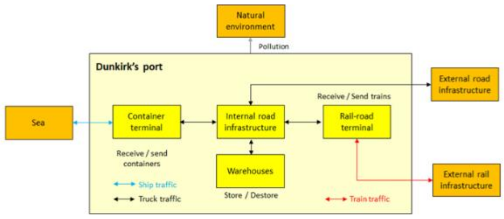
3.2.6 Synthesis: deliverables of the modelling activity

As a synthesis, we shall therefore remind that the key deliverables of the modelling activity are:

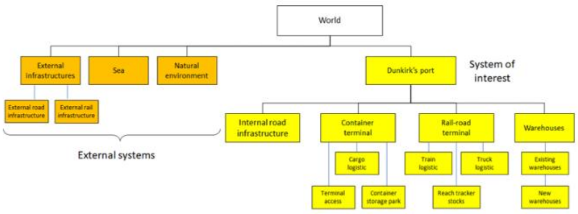
- The *areas of interest* within the system of interest,
- The *system breakdown structure* of the perimeter of interest,
- The *functional interactions diagram* of the system of interest,
- The *use cases* to analyse.

These deliverables are synthesized on Figure 17 for Dunkirk's port case.

Identification of the areas of interest within the system of interest



Main activities & interactions within the environment of the system of interest



System breakdown of the environment of the system of interest

- **Use case 1** – *Sizing of the unloading cranes fleet within the container terminal*
- **Use case 2** – *Optimization of the truck access to the container terminal*
- **Use case 3** – *Optimization of the rail-road terminal infrastructure*

Key use cases of the system of interest

Figure 17 – Key deliverables of the modelling activity of the system of interest

3.3 Activity 1.3: identifying the data to consider

To identify the **data** that one needs to consider for constructing a systemic digital twin of the system of interest, one needs to go back to the **functional interactions analysis** that provides the main flows to take into account, each flow reflecting in a corresponding set of data.

In Dunkirk's case, all these flows are especially **triggered by the import / export container flow which is associated with the ship traffic**. In other words, each container managed by Dunkirk's port, either comes initially from a ship or ends finally on a ship, meanwhile being, either transported by a truck or a train, or stored temporarily in a warehouse, as illustrated on Figure 18.

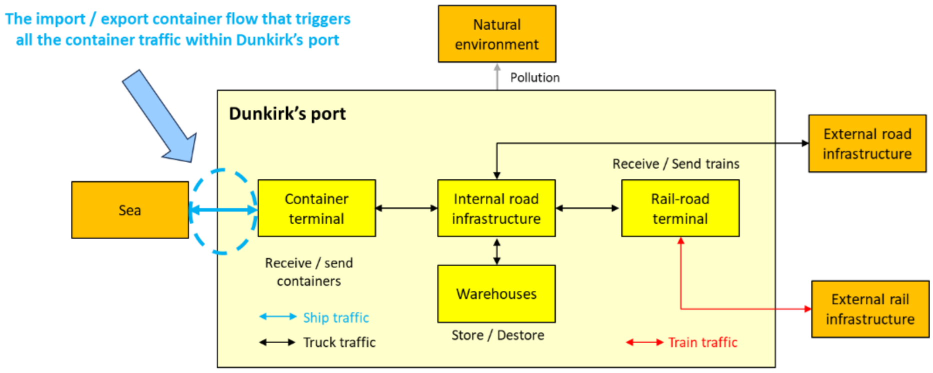


Figure 18 – The import / export container flows that trigger the traffic within Dunkirk's port

According to the previous analysis, **two main types of data**, respectively associated with the pushed imported container flow coming from the sea and the pulled exported container flow going to the sea, are required to achieve the **core of a systemic digital twin** for Dunkirk's port, that is to say:

1) the **existing & expected volumes of imported & exported container flows** managed by the container terminal,
2) the **distribution among time of these volumes** within the various systems involved in the perimeter of interest.

The below Figure 19 illustrates the corresponding data that have to be identified.

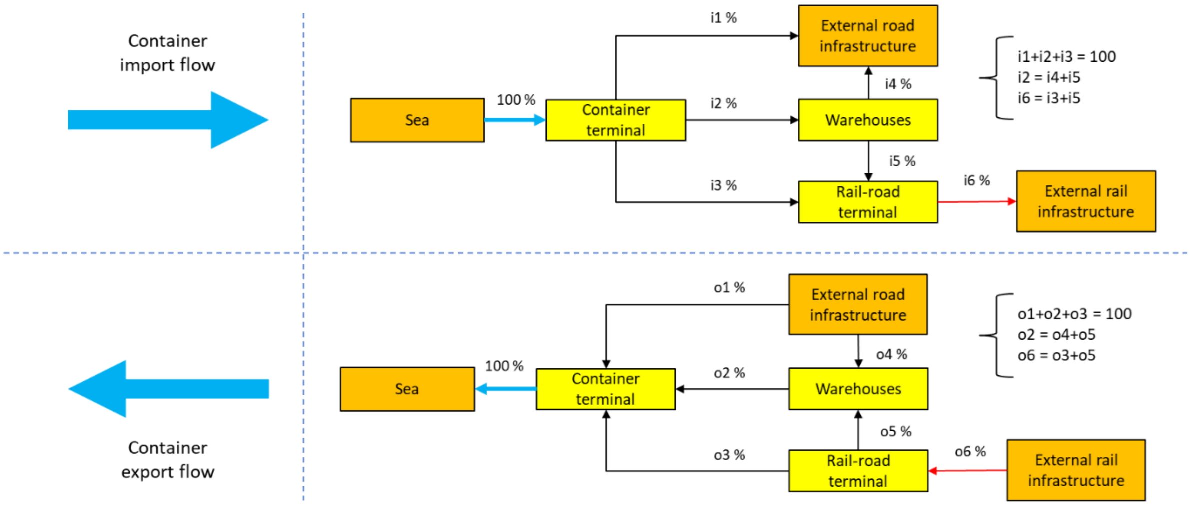


Figure 19 – The key data associated with the container flows managed by Dunkirk's port

Additional data is also required to estimate the **level of pollution** produced by the container traffic managed by Dunkirk's port (see Figure 20). This data can typically be obtained by understanding the **proportionality relationship** existing between the volume of containers managed by the port among time and such an observable.

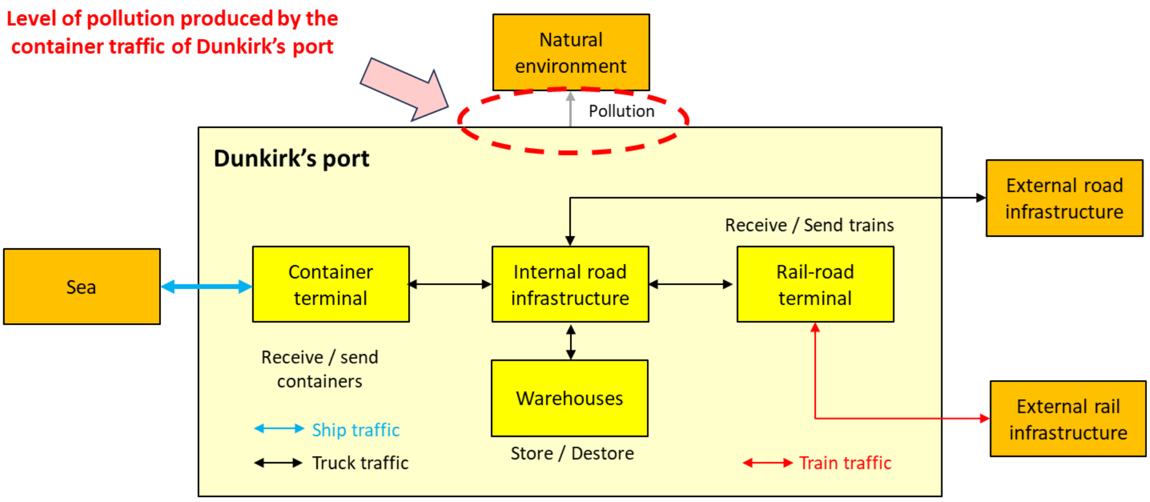


Figure 20 – The last data – related to pollution – to identify in the context of Dunkirk's port

The above table synthesizes then all the **data required to construct a systemic digital twin** of our system of interest for the Dunkirk's port case.

Flow	Data to consider
Container import flow	• Volumes of sea-managed imported containers per unit of time • Relative volumes managed within the container import network
Container export flows	• Volumes of sea-managed exported containers per unit of time • Relative volumes managed within the container export network
Pollution	• Volume of pollution produced per managed container per year

Table 2 – Data to consider in order to construct a systemic digital twin for Dunkirk's port

4. How to specify a systemic digital twin?

In order to be able to develop a systemic digital twin for an industrial system, the second phase consists in **specifying** finely a systemic digital twin, based on the material coming from the initiation phase, which starts by 2.1) defining the **simulation model architecture** and 2.2) achieving a **data analysis** in order to understand the specific data that will be used by the simulation model, resulting finally in 2.3) a **systemic digital twin specification**, as illustrated in Figure 21.

The key point to understand in this matter is the fact that the simulation model, that shall be ultimately executed, on which relies a systemic digital twin, is of course derived from the MBSE model developed in the initiation phase, but also generally quite different. One indeed needs to take here into account both computing constraints in order the simulations to be efficient and specificities of the Σ™ language that are constraining, but also possibly simplifying, the simulation model.

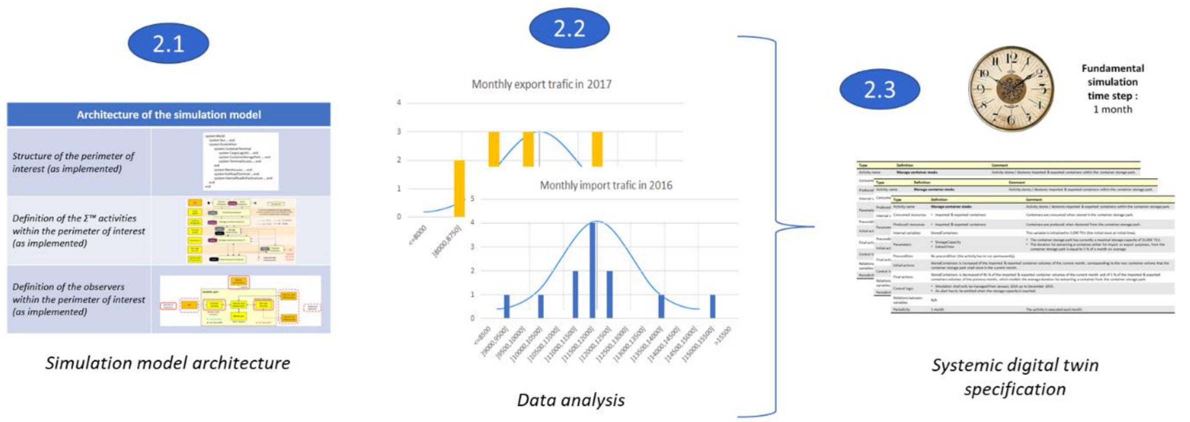


Figure 21 – Overview of the specification phase

4.1 Activity 2.1: architecting the simulation model

The very first activity of the specification phase consists in **defining the architecture** of the **simulation model** of the system of interest – to be implemented in Σ™– that shall be at the core of its targeted systemic digital twin, as ultimately generated by WorldLab™.

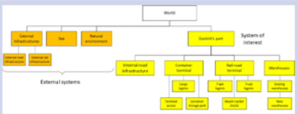
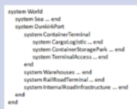
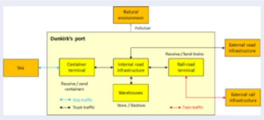
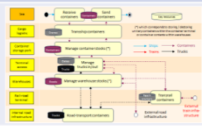
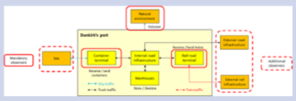
Inputs from initiation phase		Architecture of the simulation model	
System breakdown of the perimeter of interest		*Structure of the perimeter of interest (as implemented)*	
Main activities & interactions within the perimeter of interest		*Definition of the Σ™ activities within the perimeter of interest (as implemented)*	
Key performance indicators associated with the business objectives		*Definition of the observers within the perimeter of interest (as implemented)*	

Table 3 – Connexions between the inputs from the initiation phase and their simulation model counterparts

More specifically, one has to define here, based on the inputs of the initiation phase, the **structure**, the **Σ™ activities** (with their sequencing) and the **observers** of the **perimeter of interest**, as they shall be implemented, since implementation brings its own constraints and is usually not a copy/paste of the initial MBSE analysis, as already mentioned here. In this matter, Table 3 shows the relations that exist between the inputs provided by the initiation phase and their counterparts in terms of simulation model architecture:

- The system breakdown of the system of interest shall first reflect in the structure of perimeter of interest, as described in Σ™,
- The main activities & interactions within the perimeter of interest provided by its functional interaction diagram reflect then in the precise definition & sequencing of the Σ™ activities,
- Finally, the key performance indicators associated with the business objectives to fulfil shall reflect in the definition of observers in the Σ™ meaning.

The very first part of the specification phase consists therefore in **identifying the activities** – in the Σ™ meaning – and the **key resources** that they are managing / consuming, as associated with the different systems of the perimeter of interest, **together with their sequencing** within the simulation model. In this last matter, one shall especially remember that the simulation engine will repeat these activities for each chosen time step, up to the scheduled end of a simulation. Note also that one already has to **arbitrate** here between introducing an activity or using an observer, to model each part of the initial MBSE model, as defined during the initiation phase.

In Dunkirk's port case, the Σ™ activities deal first with the **containers**, that is to say receiving / sending containers from / to the sea, transhipping them through the cargo logistics, managing their stocks in the container storage park and managing the stocks of goods contained in the containers at the level of the warehouses, secondly with the **trucks** that are transporting containers by managing their in / out access in the container terminal access and their road transportation within the port and finally with **railways** where one needs to transrail containers. Concerning trucks, we shall suppose that they are available when required, which allows to abstract them, excepted at the levels of the two "manage truck in/out" and "road-transport containers" activities. We synthesized these activities in the BPMN-like diagram of Figure 22 where we also highlighted all the resources (containers, cranes, gates, trucks, goods, reach trackers) that are supporting these different Σ™ activities.

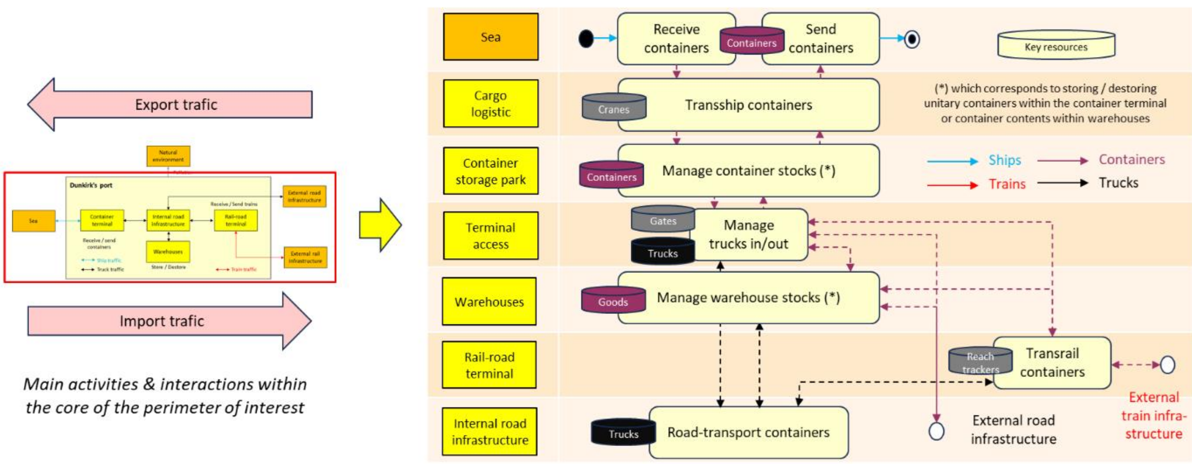


Figure 22 – Σ™ activities of the simulation model that describes the behaviour of Dunkirk's port

To define then the relevant **observers** to implement, a natural way is to associate one specific observer with each **business Key Performance Indicator (KPI)**, as initially defined, and to complete these first mandatory observers by additional ones, associated with complementary information of interest that one may possibly also need to provide.

In the Dunkirk's port case, we shall first recall that the key performance indicators to manage are the following ones:

- KPI #1: amount of new infrastructures (cranes, gates, reach trackers) required by the port
- KPI #2: customer quality of service (queuing time, volume of managed containers)
- KPI #3: level of pollution induced by the new infrastructures

As a consequence, three observers – in the Σ™ meaning – can therefore be defined to measure these key performance indicators as illustrated on Figure 23.

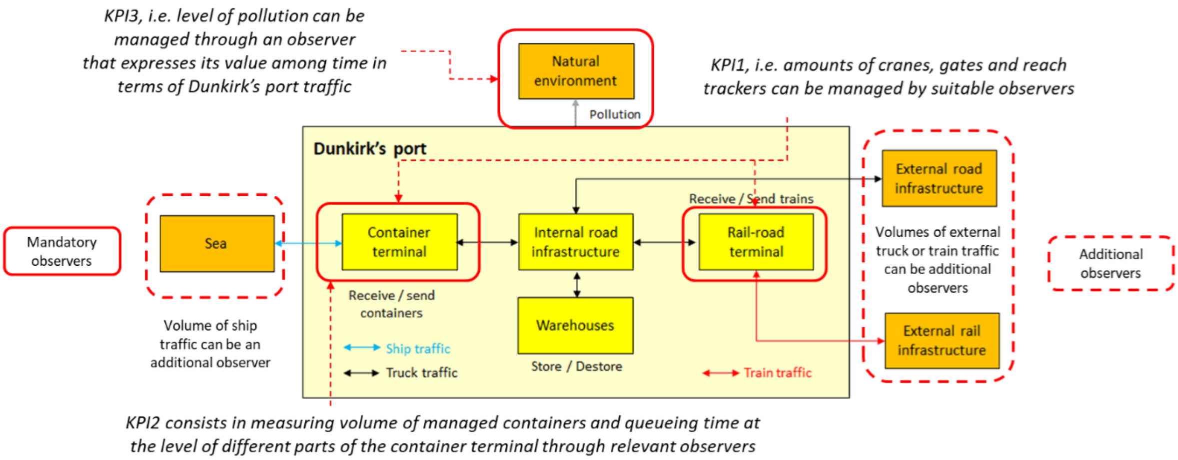


Figure 23 – Observers measuring the key performance indicators associated with Dunkirk's port

The last step of the definition of the architecture of the simulation model consists in defining the **structure of the perimeter of interest**, as implemented in Σ™, by only considering the systems which are associated with the implemented activities, and not the ones that can be covered by observers, within the system breakdown structure obtained in the initiation phase.

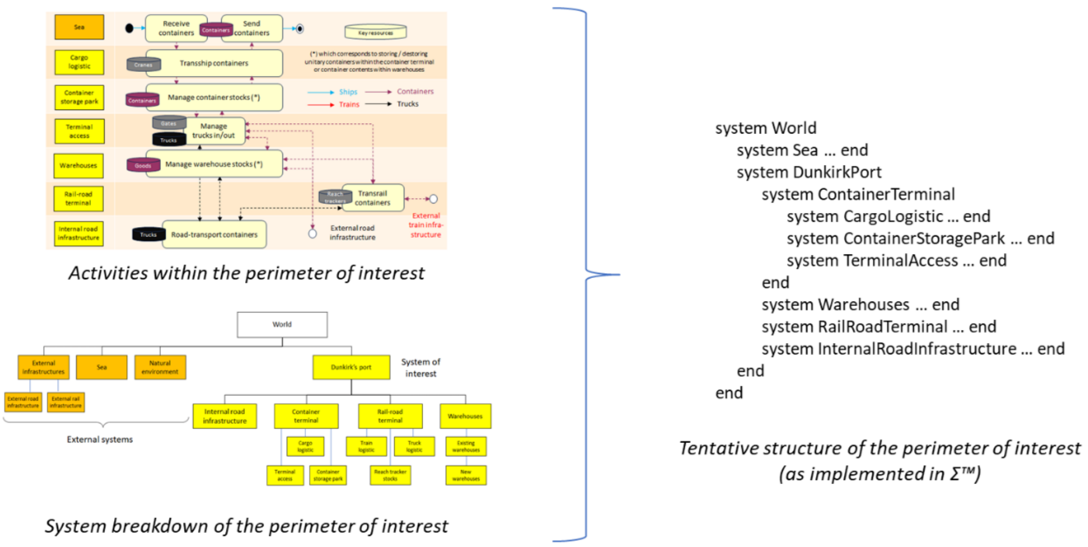


Figure 24 – The structure of the perimeter of interest for Dunkirk's port as implemented in Σ™

In order to illustrate this last step, Figure 24 provides the fragment of Σ™ that describes the structure of the perimeter of interest for the Dunkirk's port case, as it results from all previous implementation choices that were introduced here above.

4.2 Activity 2.2: analyzing the business data

The second activity of the specification phase consists in **analysing the business data** – which requires their preliminary capture, that appears in practice to be always a rather complicated activity – in order to identify the **probabilistic laws**, if any, that govern them and the main **dimensioning relationships** that are key for implementing a simulation model.

This data analysis activity is highly specific to each application case. In Dunkirk's case, it consists in analysing the **import / export past & future container volumes**, capturing the **relative containers volumes** that are managed by its logistic network and identifying the dimensioning factors associated with the **mandatory observers**, as synthesized in Figure 25. The corresponding data analysis process is presented here below, but due to confidentiality constraints, we were obliged to use **fake data**, which however does not affect the realness of the process.

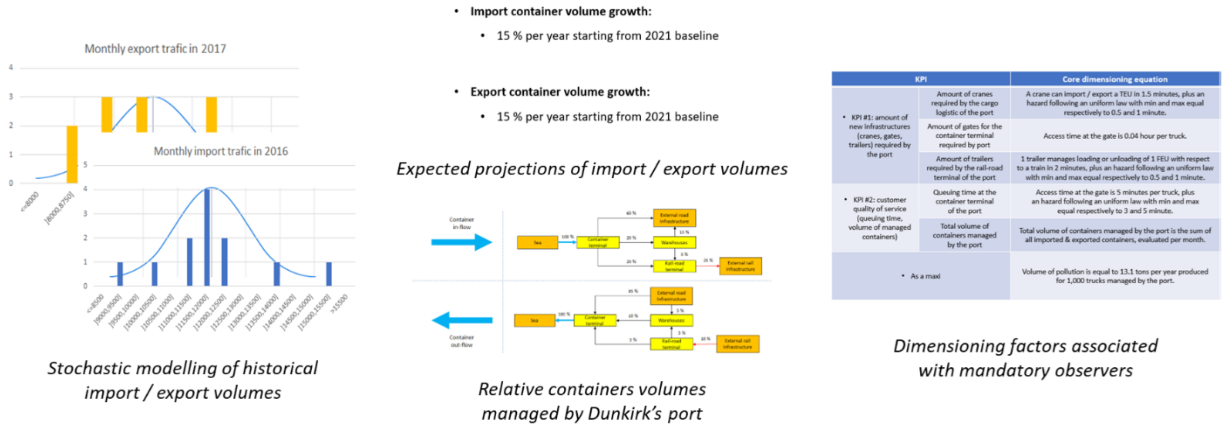


Figure 25 – Main business data analyses in Dunkirk's port case

The analysis of **imported container volume data** for the Dunkirk's port showed first that all monthly imported traffics of a given year are distributed according to **Normal laws,** whose means & standard deviations evolve among time in the same way, i.e. proportionally to the port traffic growth, since the observed ratio between these means & standard deviations remains rather stable.

In this last matter, we shall remember that a Normal law $\mathcal{N}(\mu; \sigma)$ of mean μ and standard deviation σ is a random variable $N(\mu; \sigma)$, which is characterized by the following probability law:

$$P(\, a \leq N(\mu;\sigma) \leq b \,) = \frac{1}{\sqrt{2\pi}\,\sigma} \int_a^b \frac{\exp(-(x-\mu)^2)}{\sigma^2}\, dx \,,$$

whose density has a classical bell-shape (see Figure 26 and Figure 27 for examples). The distribution of monthly container volumes in a given year y follows therefore a Normal law of mean μ and standard deviation σ if the following approximation relation is statistically valid:

$$\frac{\text{Cumulated amount of containers from january to month m within year y}}{\text{Total amount of containers for year y}} \approx P\,(\,1 \leq N(\mu;\sigma) \leq m\,)\,.$$

This last property can be observed on Figure 26 which gives the historical imported container monthly volumes for the period 2016 – 2021, which were provided by Dunkirk's port direction, with indication of the associated means, standard deviations and mean to standard deviation ratios. Data analysis especially showed that the different distributions of imported container volumes do follow a Normal law for each year involved in our set of data, as confirmed by a χ^2 test. An example of the fit between the actual data and their Normal law modelling is shown on Figure 26 for the 2016 imported container volumes, which can be captured by a Normal law of mean 12,000 and standard deviation 1,500.

Import	2016	2017	2018	2019	2020	2021
January	12 500	13 750	13 750	16 000	14 000	24 750
February	15 500	12 250	13 250	15 000	16 000	22 250
March	11 500	10 750	14 750	17 500	18 500	15 250
April	11 500	16 250	16 750	17 500	16 000	20 250
May	12 000	13 750	17 250	17 000	18 000	20 250
June	14 000	14 250	15 750	15 500	13 000	20 250
July	12 000	11 250	13 250	19 000	21 000	21 750
August	12 250	15 750	14 750	18 000	14 000	28 250
September	10 500	13 750	13 750	13 000	18 500	18 750
October	12 000	13 750	19 250	17 500	19 000	20 250
November	9 500	12 250	12 750	12 500	20 000	22 250
December	12 000	10 750	12 250	16 500	22 500	24 250
Total	145 250	158 500	177 500	195 000	210 500	258 500
Growth	N/A	9%	12%	10%	8%	23%
Mean	12 000	13 000	15 000	16 500	17 500	21 500
Standard deviation	1500	2000	2000	2000	3000	3500
Mean / standard deviation	8	7	8	8	6	6

Covid 2019 crisis impact

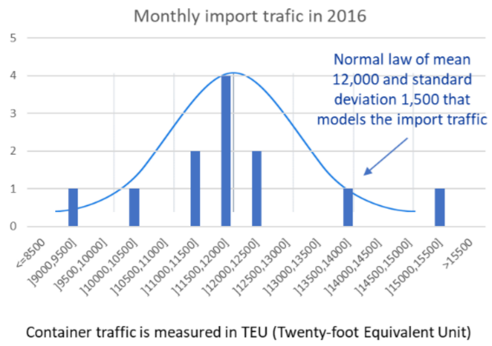


Figure 26 – Data analysis of the imported container volumes in Dunkirk's port case

In the same way, the analysis of the **exported container volume data** for the Dunkirk's port showed that the monthly exported traffics of a given year are also distributed according to **Normal laws** whose means & standard deviations evolve among time in the same way, i.e. proportionally to the port traffic growth, since observed ratios between these means & standard deviations remain again quite stable, as one can see on Figure 27, which synthesizes our data analysis for exported container volumes on the set of data provided by Dunkirk's port for the same period of time – that is to say 2016 – 2021 – than the corresponding volumes for imported containers.

Export	2016	2017	2018	2019	2020	2021
January	10 500	9 250	11 500	13 000	10 000	17 250
February	9 500	9 750	15 000	12 000	16 000	13 750
March	7 500	8 750	8 500	11 000	12 500	18 750
April	11 000	11 250	9 500	9 000	13 000	18 250
May	8 500	9 250	11 000	14 000	11 000	14 250
June	7 500	11 750	12 000	12 500	13 500	16 750
July	9 500	9 750	10 500	10 000	12 500	19 750
August	7 500	9 250	8 500	16 000	10 500	19 250
September	11 000	11 250	10 000	12 500	13 500	20 250
October	11 500	12 250	9 500	10 500	10 000	20 750
November	8 500	10 250	14 500	9 000	14 000	14 250
December	9 500	8 750	12 000	14 000	13 500	16 250
Total	112 000	121 500	132 500	143 500	150 000	209 500
Growth	N/A	8%	9%	8%	5%	40%
Mean	9 500	10 000	11 000	12 000	12 500	17 500
Standard deviation	1 500	1 000	2 000	2 000	2 000	2 500
Mean / standard deviation	6,5	10	5,5	6	6,5	7

Covid 2019 crisis impact

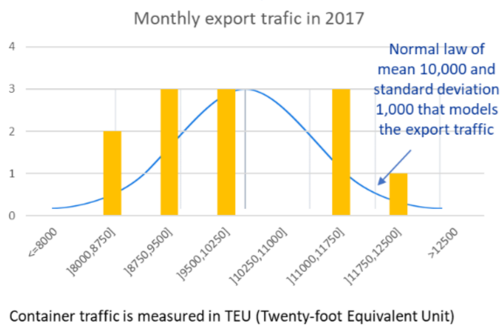


Figure 27 – Data analysis of the exported container volumes in Dunkirk's port case

One shall now point out that the main finding of the data analysis of the historical imported & exported container volumes is that the imported (resp. exported) container volumes are following Normal laws

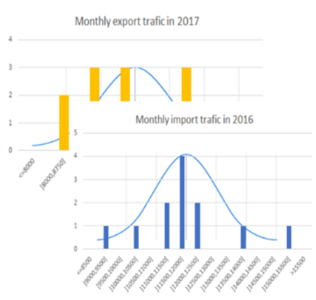
$\mathcal{N}(\mu; \sigma)$, where μ/σ can be considered as quite stable. This means than, in a first approximation, one can simulate, based on their Normal modelling, the imported or the exported container volumes for a given year knowing only their mean μ and the value of the stable ratio μ/σ. As we will see here below, we shall use this last pattern, which appeared to be valid for all years of the period 2016 – 2021, in order to simulate the evolution of the import / export traffics of Dunkirk's port in the future.

For the future container volumes managed by Dunkirk's port, we shall indeed assume that they will have a **constant growth of 15 % per year up to 2035**, symmetrically both for the imported & exported container traffics, starting from **year 2021 baseline**, that distributes per month according, as we just mentioned, to the probabilistic pattern that we identified through the previous historical data analysis. Note that these values for the future traffic growth may however be considered as parameters of our systemic digital twin in order to construct various evolution scenarios for Dunkirk's port.

- **Import container volume growth:**
  - 15 % per year starting from 2021 baseline
- **Export container volume growth:**
  - 15 % per year starting from 2021 baseline

Figure 28 – Previsions of container traffic growth in Dunkirk's port case

The understanding of the probabilistic behaviours in the past, based on historical data analysis, allows us indeed to **model the probabilistic behaviours for the future**, that shall be made according to the probabilistic Normal pattern observed in the past, taking into account the projected evolution of the future business data. Here, the baseline imported & exported container volumes are following known Normal laws in 2021, which are used to construct similar Normal laws for the imported & exported container volumes for any year n between 2022 and 2035, by considering that their means and their standard deviations are obtained by increasing each year their 2021 baseline values by a multiplicative constant $1+\gamma$ reflecting the growth γ of the traffic per year, which captures both the observed Normal law pattern in the past and the fact that the observed mean to standard deviation ratios were rather stable within the historical data. Figure 29 synthesizes this stochastic modelling approach.



Stochastic modelling of historical import / export volumes

- **Import container volume in 2021:**
  - $\mathcal{N}(21{,}500; 3{,}500)$
- **Export container volume in 2025:**
  - $\mathcal{N}(17{,}500; 2{,}500)$

Stochastic modelling of import / export volumes for baseline year 2021

- **Import container volume in 2021:**
  - $\mathcal{N}(21{,}500 \text{ x } (1+\gamma)^{(n-2021)}; 3{,}500\text{x } (1+\gamma)^{(n-2021)})$
- **Export container volume in 2025:**
  - $\mathcal{N}(17{,}500\text{x } (1+\gamma)^{(n-2021)}; 2{,}500\text{x } (1+\gamma)^{(n-2021)})$

Stochastic modelling of import / export volumes for year n $\in$ [2022, 2035]

Figure 29 – Probabilistic modelling of the traffic growth in Dunkirk's port case

The **historical relative container volumes** – which were relatively stable in the past – within the logistic network of Dunkirk are now presented in Figure 30. We shall suppose here that **these values will not significantly evolve in the future**. Note that the values for the future may however also be considered as parameters of our systemic digital twin in order to construct evolution scenarios for Dunkirk's port.

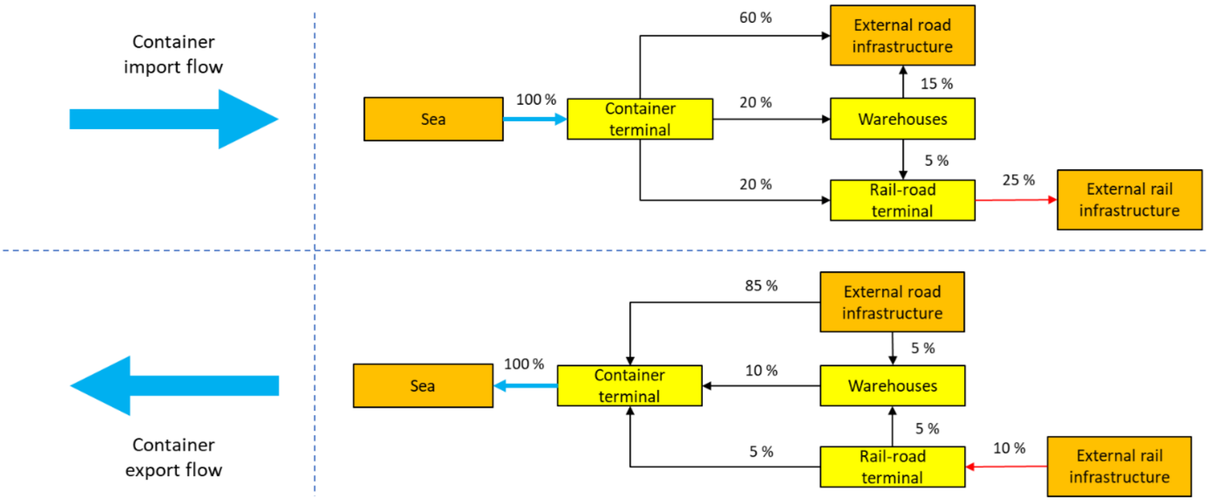


Figure 30 – Distribution of the import & export traffic within the Dunkirk's port logistic network

Finally the last step of the data analysis activity consists in identifying all dimensioning values that are used by the various observers that were defined in the previous activity (see Figure 23). In this matter, Table 4 shows for instance all numerical values, with the corresponding business explanations, that are required to precisely specify the mandatory observers associated with Dunkirk's port case.

KPI		Core dimensioning equation	Other related factors
• KPI #1: amount of new infrastructures (cranes, gates, reach trackers) required by the port	Amount of cranes required by the cargo logistic of the port	A crane can import / export a TEU in 1.5 minutes, plus an hazard following an uniform law with min and max equal respectively to 0.5 and 1 minute.	Opening time of the port is 24 hours / 24 per day on a basis of 7 days / week.
	Amount of gates for the container terminal required by port	Access time at the gate is 1.5 minutes per truck, plus an hazard following an uniform law with min and max equal respectively to 0 and 0.5 minute.	1 truck is able to manage 1.5 TEU in average.
	Amount of reach trackers required by the rail-road terminal of the port	1 reach tracker manages loading or unloading of 1 FEU with respect to a train in 2 minutes, plus an hazard following an uniform law with min and max equal resp. to 0.5 and 1 minute.	A train can handle 40 TEU A train can stop at most 4 hours in the port.
• KPI #2: customer quality of service (queuing time, volume of managed containers)	Queuing time at the container terminal of the port	Access time at the gate is 1.5 minutes per truck, plus an hazard following an uniform law with min and max equal respectively to 0 and 0.5 minute.	Queuing time shall be always under 1 hour.
	Total volume of containers managed by the port	Total volume of containers managed by the port is the sum of all imported & exported containers, evaluated per month.	The container storage park has a maximal capacity of 15,000 TEU.
• KPI #3: level of pollution induced by the new infrastructures		Volume of pollution is equal to 13.1 tons per year produced for 1,000 trucks managed by the port.	1 truck is able to manage 2 TEU in average.

Table 4 – Dimensioning factors associated with mandatory observers

4.2 Activity 2.3: specifying the systemic digital twin

We are ready to pass to the last activity of the specification phase which consists in finely specifying the target digital twin that one has to develop. The very first step of this new activity shall always be the definition of the **fundamental time step** used by the simulation engine at the core of a digital twin.

The choice of this time step is indeed key since it clearly induces the very nature of the relationships that are specifying the behaviors of the business processes of the system of interest that one shall describe in Σ™ through dedicated activities (in the Σ™meaning).

Fundamental simulation time step :
1 month

Due to the fact that all our traffic data are provided per month, we shall for instance choose to fix the **fundamental simulation time step** of our simulation model for Dunkirk's port to **1 month**, which means that each Σ™ activity shall be modelled in this case with a duration of 1 month.

This first step being done, one can now move to the **fine specification of all activities** – in the Σ™ meaning – which are involved in the **simulation model** that one has to achieve, as identified previously through its initial architecting (see the initial section of the current chapter).

To this aim, each **Σ™ activity** within the simulation model of the system of interest shall be precisely specified according to a pattern consisting in the following attributes to elicit:

- **Name:** it designs the unique identifier of the Σ™ activity.
- **Resources:** resources are modelling the physical objects and/or the information managed – i.e. consumed or produced – by the considered Σ™ activity.
- **Internal variables:** internal variables are "owned" by the Σ™ activity and linked to the system that implements the considered activity. They cannot be modified by other Σ™ activities. They often correspond to attributes associated with resources.
- **Parameters:** parameters refer to constants defined before launching a simulation. Parameters are used to characterize a simulation scenario and are linked to the system associated with the Σ™ activity to which they refer.
- **Precondition:** a precondition is a predicate, or in other words a Boolean expression, possibly complex, that captures the logical conditions which are necessarily required for the considered Σ™ activity to start.
- **Initial (resp. final) actions:** initial (resp. final) actions are executed when the Σ™ activity starts (resp. ends). They consist in consuming or producing resources and /or transforming the internal or external variables managed by the considered Σ™ activity.
- **Control logic:** control logic refers to the logical expressions which are defining – based on the internal & external variables and parameters manipulated by the Σ™ activity – under which conditions the resources and the internal & external variables manipulated by the considered Σ™ activity are modified.
- **Relations:** relations here especially refer to the analytical mathematical expressions that are defining how resources, internal & external variables and parameters, which are manipulated by the considered Σ™ activity, are related to each other.
- **Periodicity:** periodicity is the duration – in a given time unit – that separates the start and the end of the considered Σ™ activity.

Table 5 shows for instance an example of such specification of an activity in the Σ™ meaning – here the "Manage container stocks" activity – in the context of the Dunkirk's port case study.

Type	Definition	Comment
Activity name	**Manage container stocks**	Activity stores / destores imported & exported containers within the container storage park.
Consumed resources	• Imported & exported containers	Containers are consumed when stored in the container storage park.
Produced resources	• Imported & exported containers	Containers are produced when destored from the container storage park.
Internal variables	StoredContainers	This variable is initialized to 1,000 TEU (the initial stock at initial time).
Parameters	• StorageCapacity • ExtractTime	• The container storage park has currently a maximal storage capacity of 15,000 TEU. • The duration for extracting a container, either for import or export purposes, from the container storage park is equal to 20 % of a month on average
Precondition	No precondition (the activity has to run permanently).	
Initial actions	StoredContainers is increased of the imported & exported container volumes of the current month, corresponding to the new container volume that the container storage park shall store in the current month.	
Final actions	StoredContainers is decreased of 80 % of the imported & exported container volumes of the current month and of 20 % of the imported & exported containers volumes of the previous month, which models the average duration for extracting a container from the container storage park.	
Control logic	• Simulation shall only be managed from January 2016 up to December 2035. • An alert has to be emitted when the storage capacity is reached.	
Relations between variables	N/A	
Periodicity	1 month	The activity is executed each month.

Table 5 – Example of an activity specification in Dunkirk's port case

5. How to develop a systemic digital twin?

In order to construct a systemic digital twin for an industrial system, the third phase focuses on the concrete **development** – using the WorldLab™ platform – of the systemic digital twin of the system of interest, based on the material coming from the specification phase: it consists in 3.1) implementing the core **Σ™ model** on which relies the systemic digital twin and 3.2) its **user interface** through a WIDL specification, including 3.3) **verification & validation** activities (see Figure 31).

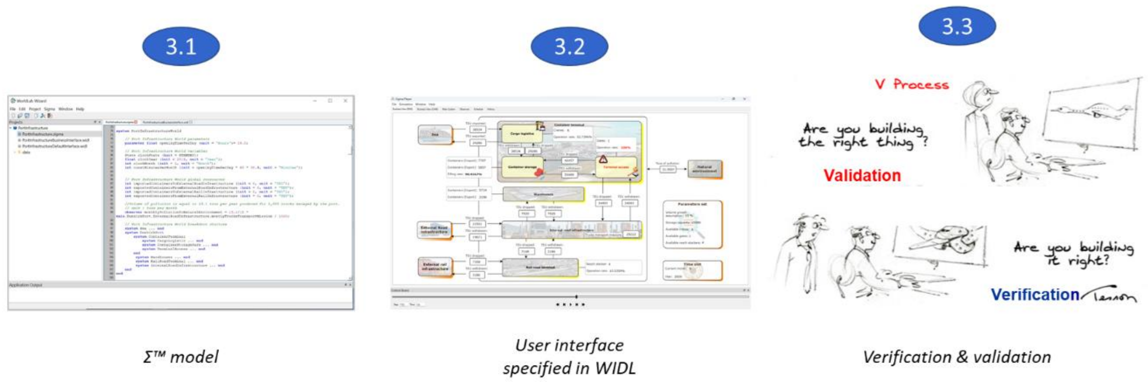


Figure 31 – Overview of the development phase

5.1 Activity 3.1: developing the core Σ™ model of the systemic digital twin

The very first activity of the **development** of the systemic digital twin of the system of interest consists in implementing the core **Σ™ model** on which relies the systemic digital twin, based on the inputs of the specification phase. Note however that this activity can possibly lead to specification changes when specification ambiguities are elicited & resolved during implementation (see Figure 32).

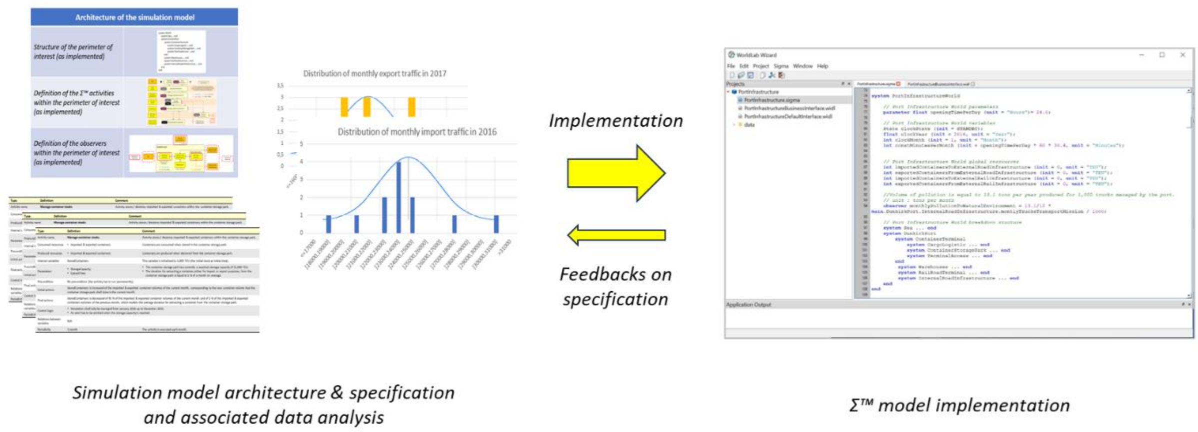


Figure 32 – Developing the core Σ™ model of the systemic digital twin

According to the very nature of the Σ™ language (see appendix A), the resulting Σ™ model is organized in two main parts: the first one – quite short – describes the hierarchical structure of the perimeter of interest, here called "PortInfrastructureWorld", when the second one – much more detailed – provides the specification of the business processes – implemented through so-called activities in Σ™ – that are managed by the various systems involved in the perimeter of interest. These two parts are illustrated on Table 6, where the left-hand side presents how the description of the structure of the perimeter of interest was implemented in Σ™ and the right-hand side shows the implementation in Σ™ of one transverse process, here time management, implemented through a clock that monitors, on a monthly basis, our simulations, that are all starting in January 2016 and ending in December 2035.

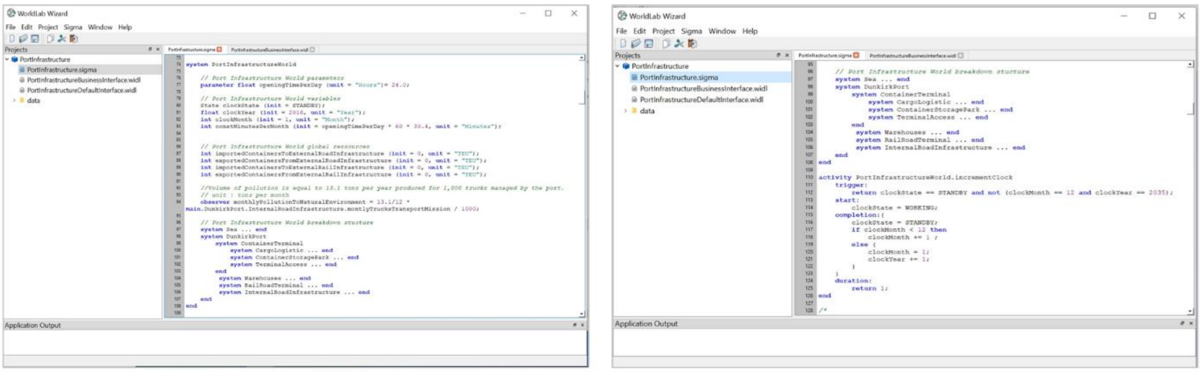


Figure 33 – Extract of the Σ™ model implementation for the Dunkirk's port case

The details of the structure of the perimeter of interest are also presented in Table 6, where one can see all the global constants and variables used in our Σ™ model, but also how the pollution indicator is implemented with a global observer "monthlyPollutionToNaturalEnvironment", which just expresses the corresponding mathematical relationship as provided by Table 4.

```
/*
 * 2. Port Infrastructure World
 * ------------------
 */

system PortInfrastructureWorld

    // Port Infrastructure World parameters
    parameter float openingTimePerDay (unit = "Hours")= 24.0;

    // Port Infrastructure World variables
    State clockState (init = STANDBY);
    float clockYear (init = 2016, unit = "Year");
    int clockMonth (init = 1, unit = "Month");
    int constMinutesPerMonth (init = openingTimePerDay * 60 * 30.4, unit = "Minutes");


    // Port Infrastructure World global ressources
    int importedContainersToExternalRoadInfrastructure (init = 0, unit = "TEU");
    int exportedContainersFromExternalRoadInfrastructure (init = 0, unit = "TEU");
    int importedContainersToExternalRailInfrastructure (init = 0, unit = "TEU");
    int exportedContainersFromExternalRailInfrastructure (init = 0, unit = "TEU");

    //Volume of pollution is 13.1 tons per year for 1,000 trucks managed by the port.
    // unit : tons per month
    observer monthlyPollutionToNaturalEnvironment = 13.1/12 *
main.DunkirkPort.InternalRoadInfrastructure.montlyTrucksTransportMission / 1000;

    // Port Infrastructure World breakdown stucture
    system Sea ... end
    system DunkirkPort
        system ContainerTerminal
            system CargoLogistic ... end
            system ContainerStoragePark ... end
            system TerminalAccess ... end
        end
         system Warehouses ... end
         system RailRoadTerminal ... end
         system InternalRoadInfrastructure ... end
    end
end
```

Table 6 – Specifying the structure of the perimeter of interest in Σ™

The details of the – quite straightforward – implementation of the clock that monitors on a monthly basis all our simulations are provided in Table 7.

```
activity PortInfrastructureWorld.incrementClock
    trigger:
        return clockState == STANDBY and not (clockMonth == 12 and clockYear == 2035);
    start:
        clockState = WORKING;
    completion:{
        clockState = STANDBY;
        if clockMonth < 12 then
            clockMonth += 1 ;
        else {
            clockMonth = 1;
            clockYear += 1;
        }
    }
    duration:
        return 1;
end
```

Table 7 – Specifying in Σ™ the clock that manages the simulation time

As a last example, it is also interesting to provide the implementation of the container importation / exportation laws in Σ™, presented in Table 8, which expresses that these laws are based on historical data up to December 2021 and on the projection formula from Figure 28, depending on the expected annual growth of the container traffic in Dunkirk's port, from January 2022 up to December 2035, all of these relations following the Normal law-based data analysis which is explained in section 4.2.

```
activity PortInfrastructureWorld.Sea.ImportExportContainers
    trigger :
        return importExportContainersState == STANDBY and main.clockState == WORKING;
    start:{
        importExportContainersState = WORKING;

        if main.clockYear < 2022 and main.clockMonth==1 then {
            //ImportedContainer concern
                meanImportContainerVolumes = empiricalDistribution("data/MeanImportContainerVolumes.tsv", step, main.clockYear);
                standardDeviationImportContainerVolumes = empiricalDistribution("data/SDImportContainerVolumes.tsv", step, main.clockYear);
            //ExportedContainer concern
                meanExportContainerVolumes = empiricalDistribution("data/meanExportContainerVolumes.tsv", step, main.clockYear);
                standardDeviationExportContainerVolumes = empiricalDistribution("data/SDExportContainerVolumes.tsv", step, main.clockYear);
        }
        if main.clockYear >= 2022 and main.clockMonth==1 then {
            //ImportedContainer concern
                meanImportContainerVolumes = round(meanImportContainerVolumes * (1.0 + growthVolumeImportedcontainer/100));
                standardDeviationImportContainerVolumes = round(standardDeviationImportContainerVolumes * (1.0 + growthVolumeImportedcontainer/100));
            //ExportedContainer concern
                meanExportContainerVolumes = round(meanExportContainerVolumes * (1.0 + growthVolumeExportedcontainer/100));
                standardDeviationExportContainerVolumes = round(standardDeviationExportContainerVolumes *(1.0 + growthVolumeExportedcontainer/100));
        }
        //ImportedContainer concern
            importedContainers += round(normalDeviate(meanImportContainerVolumes,standardDeviationImportContainerVolumes));
        //ExportedContainer concern
            exportedContainers += round(normalDeviate(meanExportContainerVolumes,standardDeviationExportContainerVolumes));
    }
    completion:{
        importExportContainersState = STANDBY;
    }
    duration:
        return importExportContainersDuration;
end
```

Table 8 – Specifying in Σ™ the container importation / exportation laws

5.2 Activity 3.2: designing the user interface of the systemic digital twin

The **design of the user interface** is the next crucial part of the development process since it consists in defining **what shall be seen by the business user** when using a systemic digital twin. The user interface shall indeed **reflect the business objectives & associated KPIs** and be able to **cover the business use cases** that were defined during the initiation phase (see Figure 34).

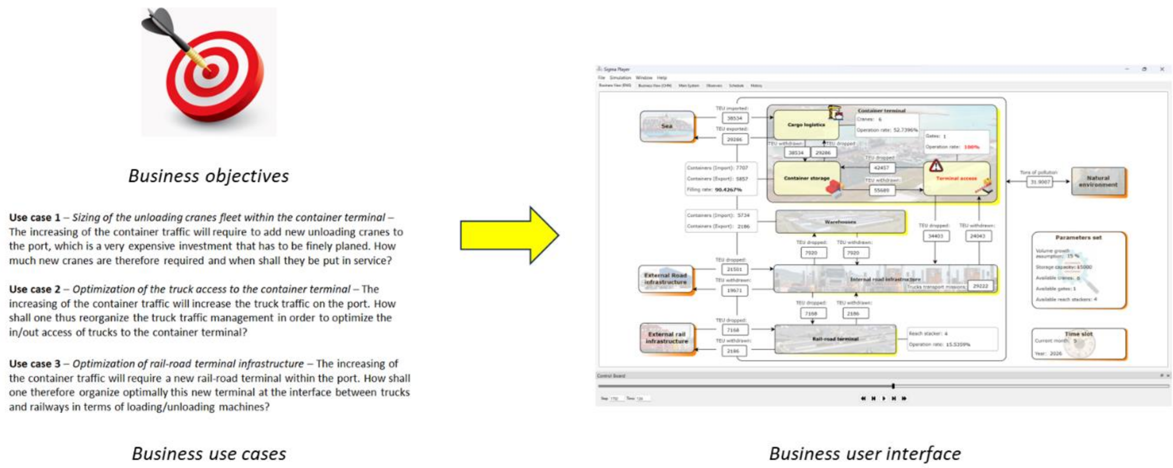


Figure 34 – Definition of the business user interface

In the context of Dunkir'ks port case, the main business user interface – for managing step-by-step simulation – will in particular look as shown in Figure 35. It shall allow the business user to simulate the evolution of the container traffic within Dunkirk's port among time, under various hypotheses such as an yearly container traffic growth assumption, the capacity of the container storage park measured in TEU, the number of available cranes for transhipping the containers, the number of access gates for trucks to the container storage park and the number of reach trackers in the rail-road terminal, which are parameters that the business user can set – their values can especially be seen on the right-hand side of Figure 35 – in order to define a business scenario to simulate. The scenario which is simulated in Figure 35 corresponds for instance to the situation where one is using the existing infrastructures of Dunkirk's port as they are and one can see that they saturate in May 2026, which means that one shall invest in order to resize them much earlier in the past.

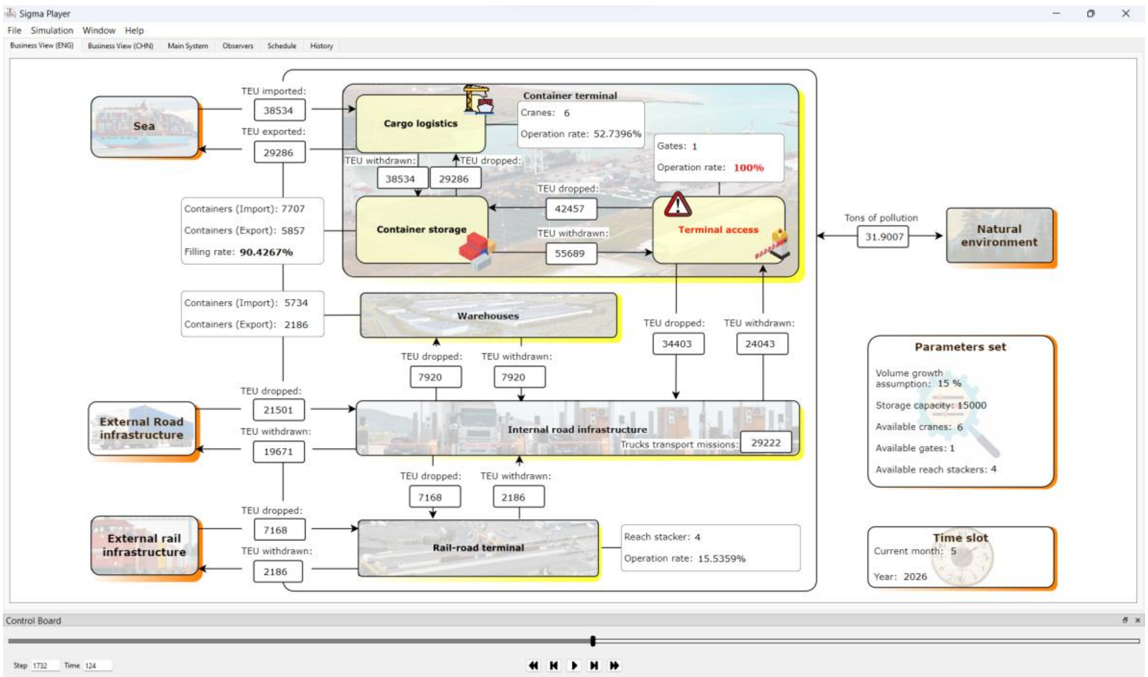


Figure 35 – The business user interface for the Dunkirk's port case (intermediate status of a simulation)

One can moreover see in Figure 36 that the existing sizing of the container terminal of Dunkirk's port will ultimately lead in December 2035 to a complete saturation of all the key infrastructures (i.e., cranes, storage park, access gates) of the container terminal.

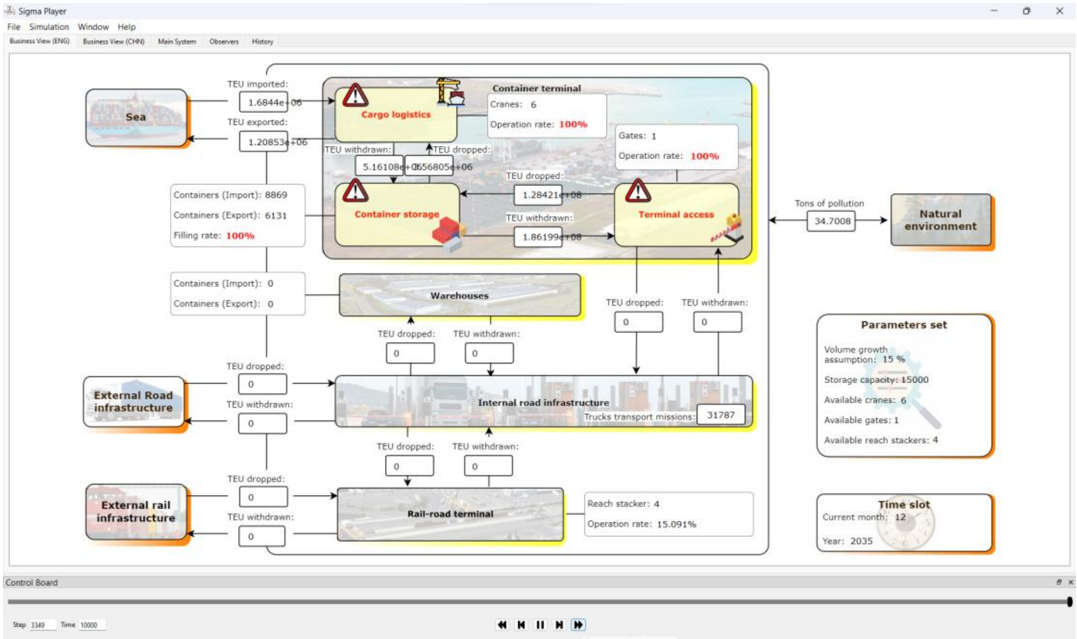


Figure 36 – The business user interface for the Dunkirk's port case (final status of a simulation)

To achieve such a business interface, one shall especially develop a WIDL model (see appendix B for more details) that describes its structure and its connection with the underlying Σ™ model by specifying the exact locations where one shall see a given value computed from the simulation of the model of the system of interest (see Figure 37 for an illustration on Dunkirk's port case).

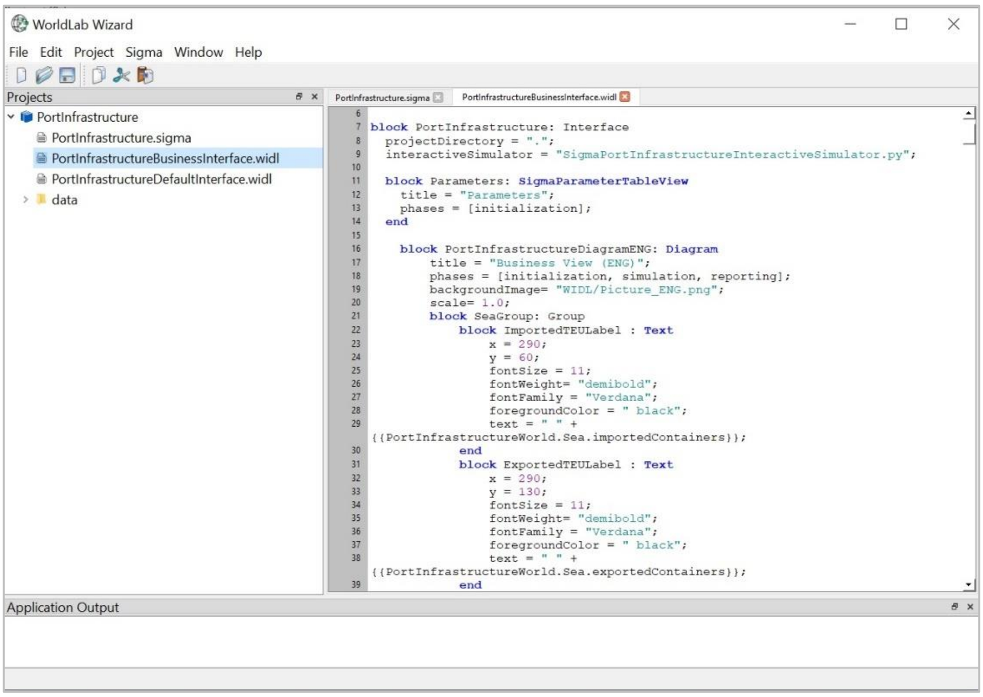


Figure 37 – The WIDL description of the business user interface for the Dunkirk's port case

5.3 Activity 3.3: verifying and validating the systemic digital twin

Last, but not least, **verification** is a process that shall be managed **permanently** along the development of a systemic digital twin and not only at its end. It consists in checking regularly 1) the **alignment between the implementation & the specification** of the simulation model, which can result in modifications of the specification due to implementation constraints and 2) the **internal consistency** of the simulation model implementation in Σ™ (see)..

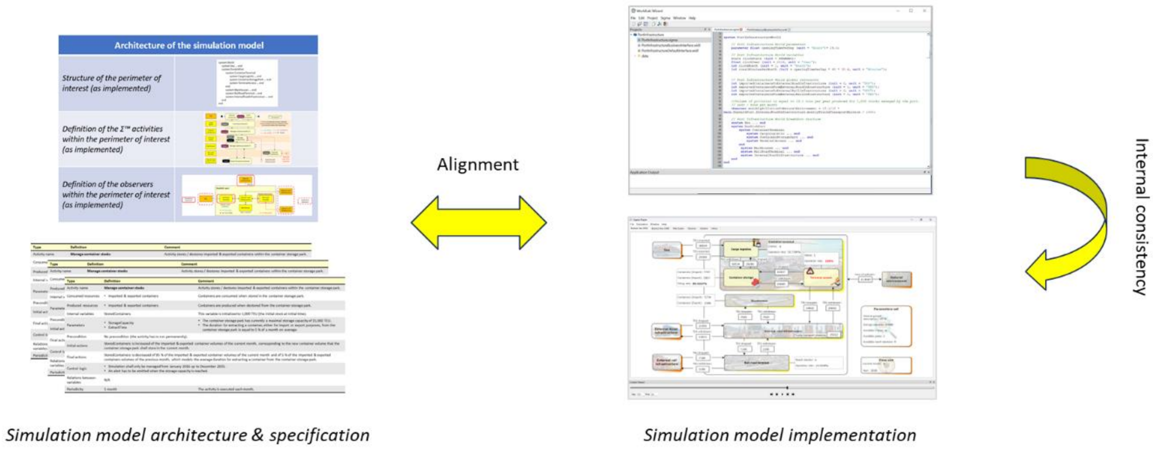


Figure 38 – Principles of verification activities

The final activity of the development of a systemic digital twin is then a validation phase, consisting in **discussing & challenging with business users & experts** the business relevance of the results that are provided by the systemic digital twin.

This last activity is fundamental and shall always be integrated in any development of a systemic digital twin, as soon as one wants to guarantee a successful and relevant development process.

Note however that validation can ultimately lead to implementation changes in order to capture as well as possible the business reality of the perimeter of interest.

6. How to use a systemic digital twin?

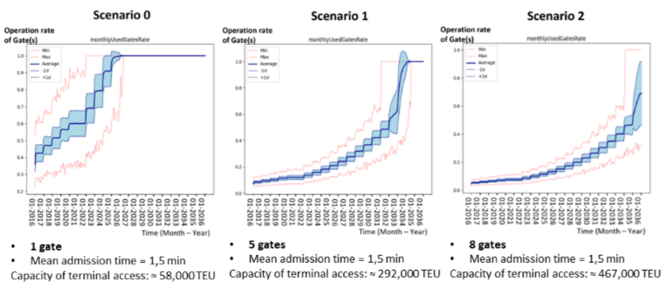
The last, but not least, phase can now deal with the **use** of the systemic digital twin of the system of interest, as actually implemented, in order to **analyze the business use cases** and contribute to the **business strategy** as defined during the initiation phase. The typical organization of this last phase consists, **for each use case** to deal with, in 4.1) identifying the **business scenarios** to analyze and 4.2) evaluating & comparing these business scenarios in order to **find the best one** (see Figure 39).

4.1

Parameters	System location	Scenario 0	Scenario 1	Scenario 2
Growth in import-export volumes after 2022	Sea	15 %	15 %	15 %
Number of cranes	Cargo logistics	6	10	13
Container storage park capacity	Container storage park	15,000	35,000	50,000
Number of gates	Terminal access	1	5	8
Number of reach stackers	Rail road terminal	1	5	4

Business scenarios associated with an use case

4.2



Evaluation & comparison of the business scenarios associated with an use case

Figure 39 – Overview of the use phase

6.1 Activity 4.1: identifying the business scenarios of each use case

The very step of the use phase consists in identifying the **business scenarios** to analyze & evaluate, based on our systemic digital twin for the perimeter of interest, for **each use case** that one shall consider as provided in the initiation phase. This activity reflects in **eliciting the parameters** – as implemented in the Σ™ model of the system of interest – that define these scenarios and **choosing the numerical values** that are characterizing each scenario.

To illustrate this new activity, let us consider **use case 2** of Dunkirk's case, which deals with *optimization of the truck access to the container terminal*, as defined during the initiation phase. In this matter, we shall remember that the admission of trucks to the container terminal is a process that is carried out manually by one gate. This process is therefore suitable today for the current flow of containers which is transported by trucks on roads, but will undoubtedly pose capacity problems in the future. At this level, the business challenges that the port is facing are especially the following ones:

- Anticipate blocking of terminal access

- Control / limit the impact on air pollution of the increase in the number of trucks that are serving the container terminal.

- Avoid loss of customers due to poor quality of service
- Avoid forwarding traffic to other ports

Three progressive business scenarios can then be associated with this last use case, under a common assumption of 15 % annual growth of the container traffic within Dunkirk's port:

- Scenario 0 is the **baseline scenario** where one does nothing, that is to say where one just reuse the existing infrastructure of Dunkirk's port, consisting in six transshipping cranes, a container storage park with capacity of 15,000 TEU, one single gate for managing the truck access of the container terminal and a rail-road terminal with one single reach tracker (to model the current minimalist situation existing Dunkirk's port);

- Scenario 1 is an **intermediate scenario** with ten transshipping cranes, a container storage park with capacity of 35,000 TEU, five gates for managing the truck access of the container terminal and a rail-road terminal with five reach trackers;
- Scenario 2 intends to be a **robust scenario** with thirteen transshipping cranes, a container storage park with capacity of 50,000 TEU, height gates for managing the truck access of the container terminal and a rail-road terminal with four reach trackers

The below Figure 40 synthesizes these three business scenarios in one single table.

Parameters	System location	Scenario 0	Scenario 1	Scenario 2
Growth in import-export volumes after 2022	Sea	15 %	15 %	15 %
Number of cranes	Cargo logistics	6	10	13
Container storage park capacity	Container storage park	15,000	35,000	50,000
Number of gates	Terminal access	1	5	8
Number of reach stackers	Rail road terminal	1	5	4

Figure 40 – The three scenarios associated with use case 2 that we shall consider

6.2 Activity 3.2: evaluating & comparing the business scenarios of each use case

The second and last activity of the use step consists finally in simulating and comparing the various scenarios identified during the previous activity, through stochastic simulations consisting in running a **huge number**, typically around 10,000, **of stories with Monte-Carlo stochastic simulations**, in order to get relevant statistical values for the key performance indicators of interest.

In the context of the analysis of the use case 2 associated with Dunkirk's port, the key performance indicator of interest is the monthly operation rate of a gate. The stochastic simulation of the baseline scenario 0 consists then in playing 10,000 simulations of the Σ™ model of the system of interest, as achieved during the development phase, with the following data:

- One single admission gate with 90 seconds processing time at the most restrictive point,
- Each truck transports in average the equivalent of 2 TEU,
- Truck arrival follows an empirical distribution corresponding to the actual observed data,
- Future logistic flows are given by Dunkirk's port growth trends.

Such stochastic simulations can be easily managed with WorldLab™ which offers this important feature to its users for evaluating key performance indicators on a given Σ™ model.

In Dunkirk's case, the first corresponding result is then provided in Figure 41 which gives the monthly operational use of the access gates of the container terminal – on a scale from 0 % to 100 % – for a period running from 2016 to 2035, integrating the evolution of the container traffic in Dunkirk's port.

The line in dark blue shows here the average value of the monthly operational use of the access gates during the simulated period, when the zone in light blue corresponds to the maximum likelihood area of this indicator, defined by minus / plus one standard deviation with respect to the previous average value, and the two lines in dashed red represent the minimal and maximal values of the indicator of interest for all simulations. One can therefore immediately see that the access gates do saturate by early 2025 in this baseline scenario, where existing port infrastructures are reused without any change, and that first saturations already begin to occur during the 2023-2024 period in extremal cases.

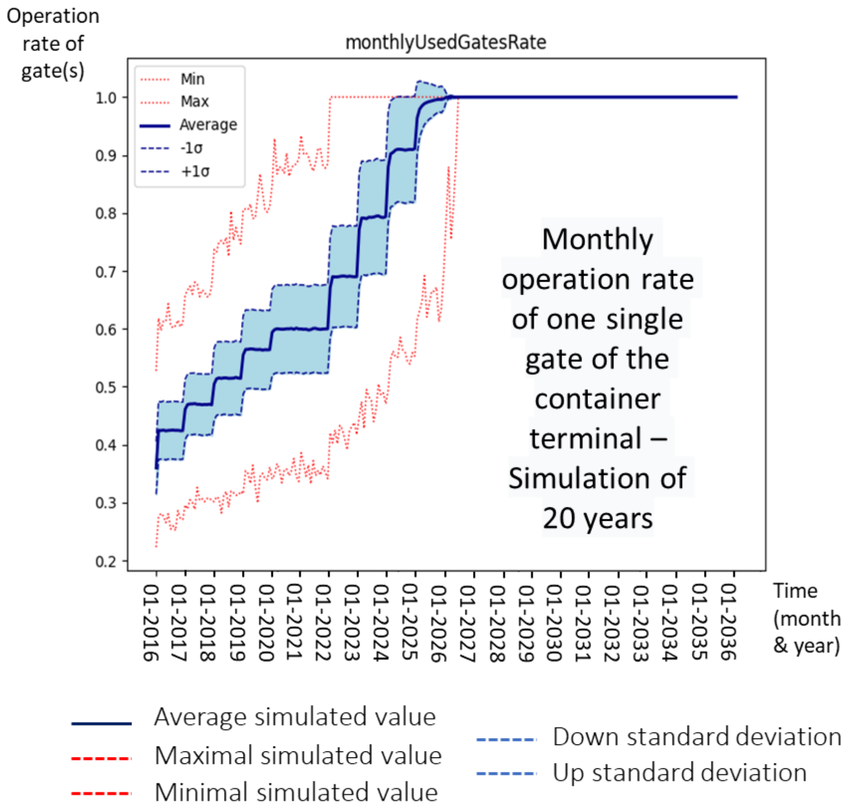


Figure 41 – Stochastic simulation of the baseline scenario 0

One can then also continue to manage with WorldLab™ the stochastic simulations (see Figure 42) that are required in order to evaluate in the same way the two other scenarios associated with use case 2 for Dunkirk's port, as presented in the previous section.

These new stochastic simulations especially show that:

- In scenario 1, access gate saturation occurs between 2031 to 2035 in 100 % of cases,
- In scenario 2, access gate saturation may occur between 2034 to 2035, but only for less than 15 % of cases.

To avoid gate saturation (therefore truck waiting time), the **most effective solution for 2035** is thus **scenario 2**, consisting in **adding 7 new access gates**, which requires managing the construction of the corresponding resources, in order to achieve a smooth admission to the container terminal. Note also that the stochastic simulation shows that saturation may still happen in scenario 2, but only in extremal situations corresponding to 15 % of the cases: the stochastic simulation shows therefore that there is a business trade-off to do here between adding more gates – which has a cost – and accepting possible saturation in a limited number of situations.

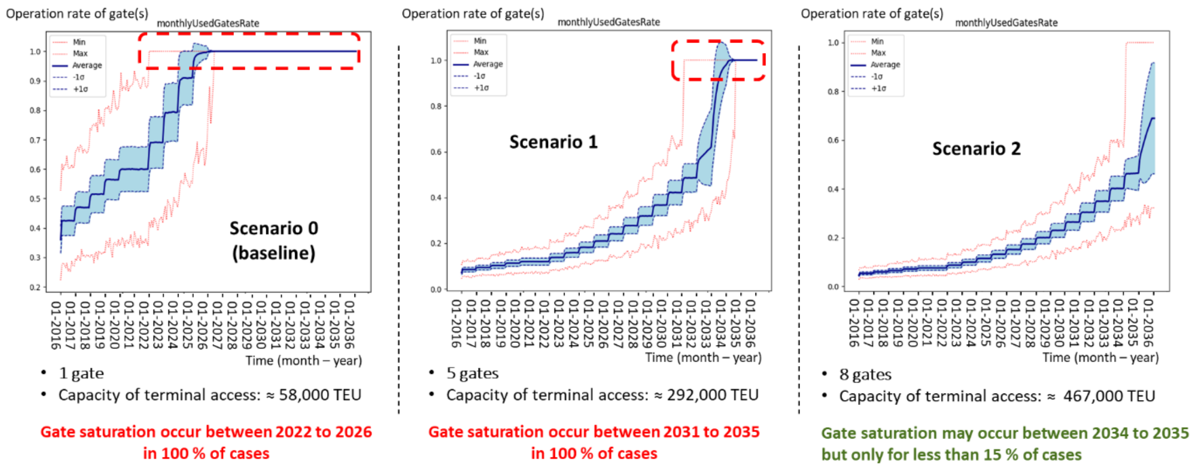


Figure 42 – Stochastic simulations of scenarios 0, 1 and 2

Appendix A: Σ™

A.1 Why creating a system specification formal language?

In order to understand better the motivation of the creation of a specific system specification formal language, let us first recall that any language can always be analyzed from three **syntax, semantics and pragmatic** perspectives. Syntax refers to the nature of the elementary symbols managed by the language, when semantics refers to the meaning of sequences of such symbols and pragmatics to the possible practical usages of these symbols in a given operational context.

Layers of a language	**Definition**	**Natural language** (e.g. Greek)	**Pseudo-formal language** (e.g. MBSE)	**Formal language** (e.g. programming language)
Pragmatics	How to use these elementary symbols in practice?		Doing lifecycle-oriented system design relying on model-based systems architecting	Developing a user-oriented software application
Semantics	What is the meaning of these elementary symbols?		Lifecycle phase	X = X0 → X = X0+1 (predicate in mathematical logic)
Syntax	What are the elementary symbols managed by the language?	ζωα	Run period	x := x+1

Table 9 – An analysis grid that works for any language

Table 9 shows examples of applications of this gris of analysis to three kinds of languages:

- A **natural language**, here Greek, where we provided a syntax consisting in the sequence of three Greek letters ζωα, whose semantics is "animals" and that can be used for instance within a theater play, like the one which is illustrated in Table 9,
- A **pseudo-formal language**, here a model-based systems engineering graphical description language such as SysML, where we presented in Table 9 a syntax formed by an oval containing the words "Run period", whose semantics is the run lifecycle phase of a system and that can for instance be used to do lifecycle-oriented system design,
- A **formal language**, i.e., a language with a sound mathematical semantics, here a programming language such as Pascal, where Table 9 shows a syntax formed of the sequence "X := X+1" of symbols, whose semantics is provided by the predicate, in the meaning of mathematical logic, "X = X0 → X = X0+1" which expresses the fact that the variable X will have the value X0+1 after processing of the considered sequence of symbols, if it initially had the value X0, which can be pragmatically used by developing for instance a user-oriented software application.

One can now use this analysis grid to compare the three main types of system specification languages that can be found in practice, as already outlined (see Table 10), that is to say:

1. The **natural languages**, which are used in practice by most of engineers to specify a system, with unfortunately a poor level of rigor and no real semantics, leading to many possible meanings & interpretations of a given specification and to the impossibility of simulating such unformal system specifications;

2. The **graphical languages**, such as BPMN, SysML or UML, based on a meta-model, which are now widely used by engineers within the model-based systems engineering (MBSE) approach: they have a better level of rigor, but which is still weak due to the absence of formal semantics that leads to structural interoperability & simulation issues,
3. The **formal languages**, such as AltaRica, based on mathematical fundamentals which equips them with a formal semantics, which are unfortunately not very used by engineers, at the exception of the safety domain: however they especially support simulation due to their strong level of rigor which suppresses any ambiguity in such a specification mode.

The key point to stress here is that only formal languages can really be simulated since being able to simulate a specification language means that there is only one – and only one – interpretation for each part of a given specification. As a consequence, the **Σ™ modeling language**, which is a **formal language** fully dedicated to **industrial system specification** on which relies our systemic digital twin approach, naturally supports **simulation** of the systems that it allows to model.

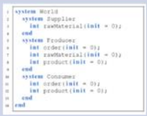
Modeling language type	Syntax used by the modeling language	Examples	Features	Fundamentals	Level of rigor	Simulation capability
Formal	Formal specification language		Formal semantics leading to compiling, simulation & strong interoperability	Mathematics	Strong	Possible
Pseudo-formal	Graphical language		No formal semantics leading to structural interoperability & simulation issues	Meta-model	Weak	Difficult since it requires a simulation semantics
Unformal	Natural language		No semantics at all leading to many possible meanings	Practice	Poor	Impossible

Table 10 – Comparing the three main types of specification languages

This is the key property that motivated the introduction of a new system specification formal language such as Σ™. Note finally that its scope of application is specifically dedicated to industrial systems within their manufacturing, operation and maintenance phases where the system behavior can be described through a **discrete-event** approach.

A.2 Σ™ follows the S2ML+X paradigm

Let us first recall that any system model has always to describe both the structure and the behavior of a system. The **S2ML+X paradigm** consists then in claiming that a **system modeling (formal) language** shall consist in the combination of a **mathematical framework** dedicated to the description of **the behavior** of the system of interest and of a **structuring paradigm** that reflects the **generic principles of organization of any system** and that is used to organize the model. This paradigm was abstracted from the safety-oriented language AltaRica developed by Antoine Rauzy in the last decades and can be synthesized in the simple "equation": *System Models = Structures + Behaviors*.

Note also that structures are largely independent of any concrete system. They can thus be described through a generic systems structure description language **S2ML**.. On the other hand, behaviors can be mathematically modeled in many different ways (e.g. through continuous or discrete modeling), the choice of the relevant **mathematical framework (X)** depending on the nature of the system of interest.

The previous “equation” transforms then in terms of specification languages into the new “equation” stating that any system specification language can be expressed according to a S2ML + X paradigm.

In this last matter, we shall now introduce the key generic constructions of the **Systems Structure Modeling Language (S2ML)** invented by Antoine Rauzy, i.e., port, connection, container, composition, aggregation, prototyping/cloning, class/instantiation and inheritance, as described in Figure 43. We refer to *Michel Batteux, Tatiana Prosvirnova, Antoine Rauzy, From Models of Structures to Structures of Models, IEEE International Symposium on Systems Engineering (ISSE 2018)*, *Roma, October 2018*, for more details on the S2ML structural modeling language.

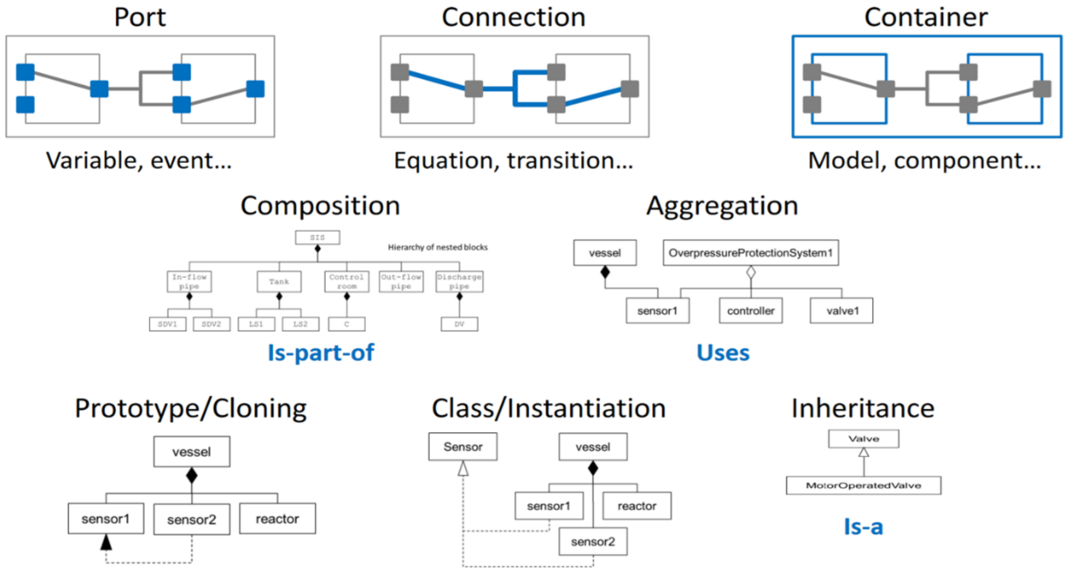
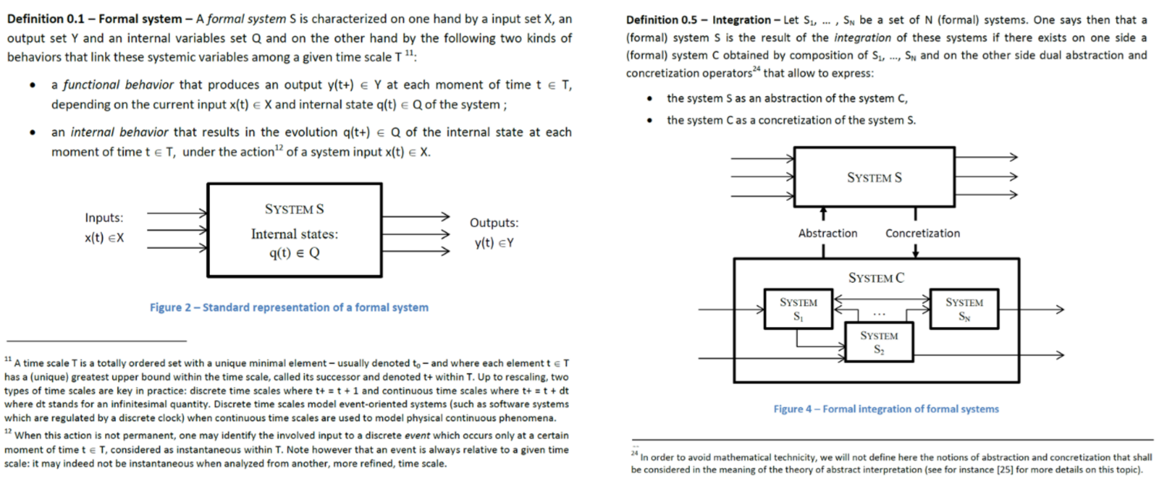


Figure 43 – The key generic constructions of the Systems Structure Modeling Language (S2ML)

It is now time to state that our **system specification formal language Σ™**, which is at the very core of our systemic digital twin approach, follows the Sys2ML + X, where the “X” is the **functional framework for a system**, described in Figure 44, applied only with a **discrete time scale**. We refer to *Daniel Krob, Model-Based Systems Architecting – How to use CESAM for architecting complex systems?, ISTE, Wiley, 2022*, for more details on that mathematical formalism.



Definition 0.1 – Formal system – A *formal system* S is characterized on one hand by a input set X, an output set Y and an internal variables set Q and on the other hand by the following two kinds of behaviors that link these systemic variables among a given time scale T [11]:

- a *functional behavior* that produces an output $y(t+) \in Y$ at each moment of time $t \in T$, depending on the current input $x(t) \in X$ and internal state $q(t) \in Q$ of the system ;
- an *internal behavior* that results in the evolution $q(t+) \in Q$ of the internal state at each moment of time $t \in T$, under the action[12] of a system input $x(t) \in X$.

Figure 2 – Standard representation of a formal system

[11] A time scale T is a totally ordered set with a unique minimal element – usually denoted t_0 – and where each element $t \in T$ has a (unique) greatest upper bound within the time scale, called its successor and denoted t+ within T. Up to rescaling, two types of time scales are key in practice: discrete time scales where t+ = t + 1 and continuous time scales where t+ = t + dt where dt stands for an infinitesimal quantity. Discrete time scales model event-oriented systems (such as software systems which are regulated by a discrete clock) when continuous time scales are used to model physical continuous phenomena.

[12] When this action is not permanent, one may identify the involved input to a discrete *event* which occurs only at a certain moment of time $t \in T$, considered as instantaneous within T. Note however that an event is always relative to a given time scale: it may indeed not be instantaneous when analyzed from another, more refined, time scale.

Definition 0.5 – Integration – Let $S_1, \dots, S_N$ be a set of N (formal) systems. One says then that a (formal) system S is the result of the *integration* of these systems if there exists on one side a (formal) system C obtained by composition of $S_1, \dots, S_N$ and on the other side dual abstraction and concretization operators[24] that allow to express:

- the system S as an abstraction of the system C,
- the system C as a concretization of the system S.

Figure 4 – Formal integration of formal systems

[24] In order to avoid mathematical technicity, we will not define here the notions of abstraction and concretization that shall be considered in the meaning of the theory of abstract interpretation (see for instance [25] for more details on this topic).

Figure 44 – The mathematical behavioral framework on which Σ™ relies (X)

A.3 The core features of Σ™

The **Σ™ modeling language** allows therefore naturally to specify the **hierarchical structure** and the **behaviors**, that is to say the business processes, of a given industrial system, as illustrated in Figure 45, but also, through the WIDL language which is presented in Appendix B, the **end-user interface** with the **business indicators & alerts** that shall be computed and shown to the business users during the use of a systemic digital twin.

The key point is that structures are specified in Σ™ in a quite intuitive way, the fact that a system can be a part of another system being expressed through the "**system** … **system** …. **end end**" construction, as illustrated on the left-hand side of Figure 45, when behaviors are specified in Σ™ through activities where one needs to explicitly define the logical condition that triggers a given activity, what shall be done when the activity starts and stops and what is the duration – in a certain unit of time – of the activity, using respectively to these different purposes the keywords "**trigger**", "**start**", "**completion**" and "**duration**", as illustrated on the right-hand side of Figure 45.

```
system World
  system Supplier
    int rawMaterial(init = 0);
  end
  system Producer
    int order(init = 0);
    int rawMaterial(init = 0);
    int product(init = 0);
  end
  system Consumer
    int order(init = 0);
    int product(init = 0);
  end
end
```

Specification of a hierarchy of systems in Σ™

```
system World.Supplier
  int rawMaterial(init = 0);
  bool renewing(init = false);
end

activity World.Supplier.RenewRawMaterialStock
  trigger:
    return rawMaterial<=1000 and not renewing;   ← Condition that triggers the activity
  start:
    renewing = true;                             ← What shall be done when the activity starts
  completion: {
    renewing = false;
    rawMaterial += 100;                          ← What shall be done when the activity stops
    }
  duration:
    return 30;                                   ← Duration of the activity (in units of time)
end
```

Specification of a business process – as an activity – in Σ™

Figure 45 – Specification of the structure and behavior of a system in Σ™

A key specificity of Σ™ is then that **stochastic behaviors** can be captured within Σ™. This can be done in two different ways, either via random variables manipulated by **activities** or via random **durations**, as illustrated in Figure 46. One can indeed express in Σ™ such stochastic behaviors either through a number of exact probabilistic distributions (e.g. Normal laws, uniform laws, exponential laws, etc.) or through empirical distributions (i.e. experimental timed sequences).

```
activity World.Consumer.ConsumeProduct
  trigger:
    return product>=1000 and not consuming;
  start:
    consuming = true;
  completion: {
    consuming = false;
    product -= uniformDeviate(1, 2.5);
    }
  duration:
    return 1;
end
```

Example of a stochastic quantity with explicit probabilistic distribution (refers to `uniformDeviate(1, 2.5)`)

```
activity World.Consumer.ConsumeProduct
  trigger:
    return product>=1000 and not consuming;
  start:
    consuming = true;
  completion: {
    consuming = false;
    product -= 1;
    }
  duration:
    return triangularDeviate(1, 5, 2);   ← Example of a stochastic duration with explicit probabilistic distribution
end
```

Figure 46 – Specification of probabilistic volumes and durations in Σ™

Stochastic simulations can be especially monitored within ΣTM in two different ways, either through **observers** which are updated continuously during the execution of a ΣTM model, or through **indicators** that are computed from observers, at certain moments of time, typically when the execution of the simulation of a given model is complete, as illustrated in Figure 47.

```
system World.Producer
  ...
  int rawMaterial(init = 0);
  ...
  observer levelRawMaterial = rawMaterial;
  ...
end
```

Declaration of a numerical observer

```
system World.Producer
  ...
  int rawMaterial(init = 0);
  ...
  observer lowLevelRawMaterial = rawMaterial<=30;
  ...
end
```

Declaration of a Boolean observer

```
system World.Producer
  indicator probabilityTooLowStock = maximum lowLevelRawMaterial(
    mean=true, standardDeviation=true);
  indicator sojournTimeTooLowStock = mean lowLevelRawMaterial(
    mean=true, standardDeviation=true, quantiles=10);
end
```

Declaration of indicators

Figure 47 – Specification of observers and indicators in ΣTM

In order to manage **systemic scenarios**, one can also especially define **parameters** in ΣTM, as illustrated in Figure 48. Each parameter corresponds to a value that can be modified by the end-user: defining a systemic scenario consists then just in defining the values of a given set of parameters, which can be done through a specific generic user interface, automatically created by the WorldLab™ platform when a given Σ™ model is compiled in order to generate the associated systemic digital twin.

```
system World.Supplier
  parameter int renewalThreshold = 1000;
  parameter int renewalAmount = 100;
  parameter int renewalDuration = 30;
  int rawMaterial(init = 0);
  bool renewing(init = false);
end

activity World.Supplier.RenewRawMaterialStock
  trigger:
    return rawMaterial<=renewalThreshold and not renewing;
  start:
    renewing = true;
  completion: {
    renewing = false;
    rawMaterial += renewalAmount;
    }
  duration:
    return renewalDuration;
end
```

Declaration of three parameters

Figure 48 – Specification of parameters in ΣTM

Last, but not least, the ΣTM language also allows to **manage deformable systems**, that is to say systems whose structure or behaviors do change among time, through different mechanisms as illustrated in

Figure 49 where we show how to add a new system to an existing one on the left-hand side and how to destruct and create a given element managed by a system on the right-hand side.

```
system World
  ...
  system Producer ... end
  ...
  system Supplier
    ...
    system RawMaterial
      ...
    end
  end
  ...
end

system World.Producer
  embeds main.Supplier.RawMaterial as RM;
  ...
  activity ProduceProduct
    trigger:
      return RM.price<priceThreshold ... ;
    ...
  end
end
```

The producer system aggregates the RawMaterial system

```
activity World.Consumer.PlaceOrder
  Order* order;
  trigger: {
    if placingOrder then return false;
    for order in Orders
      if order->status==PREPARED then return true;
    return false;
    }
  start: {
    placingOrder = true;
    Orders.remove(order)
    }
  completion: {
    order->status = PLACED;
    main.Producer.Orders.append(order)
    placingOrder = false;
    }
  duration:
    return 1;
end
```

Destruction & creation of orders within a Consumer system

Figure 49 – Specification of deformable systems in Σ™

Appendix B: WIDL

B.1 WIDL follows the S2ML+X paradigm

As Σ™, the **WorldLab™ Interface Description Language – WIDL –** is another **S2ML+X modeling language** which is dedicated to the **specification of systemic digital twin user interfaces**. Here the "X" refers to a description of the organization & contents of graphic views.

More specifically, a WIDL model describes the whole interface as structured in a series of views, as illustrated in Figure 50, which can be visible in any of the three standard phases of a simulation: **initialization phase**, i.e. before the start of the simulation; **simulation phase**, i.e. when the simulation is running; **reporting phase**, i.e. when the simulation is finished.

```
block Nautilus: Interface
  projectDirectory = "..";
  interactiveSimulator = "SigmaNautilusInteractiveSimulator.py";
  block Parameters: SigmaParameterTableView
    title = "Parameters";
    phases = [initialization];
  end
  block MainSystem: SigmaSystemListView
    title = "Main System";
    phases = [initialization, simulation, reporting];
    filter = [systems, parameters, variables, observers];
  end
  block History: SigmaHistoryTableView
    title = "History";
    phases = [simulation, reporting];
    length = 20;
  end
end
```

Whole interface

Views forming the interface

Figure 50 – Example of WIDL specification

B.2 The core features of WIDL

WIDL models involve three **core objects**: blocks, variables & expressions as illustrated in Figure 51.

Identifier of a block *Class*

```
block Nautilus: Interface
  …
  block MainSystem: SigmaSystemListView
    title = "Main System";
    phases = [initialization, simulation, reporting];
    filter = [systems, parameters, variables, observers];
  end
  …
end
```

Identifier of a variable *Expression*

Figure 51 – Blocks and variables in WIDL

On one hand, **blocks** are containers for other objects, including blocks. All graphical objects, from the whole interface to a basic element of a graphic diagram, are represented by blocks. A block has an identified and a class which is either a basic class or a user-defined class. On the other hand, **variables** have an identifier and a value which is an **expression**.

WIDL offers then a number of **generic views** that we shall now present on a one-by-one mode.

- *View 1 - parameter table*: a **parameter table**, whose basic class is **SigmaParameterTableView**, displays **all parameters of a ΣTM model**, so that one can modify their values before launching a simulation, as illustrated in Figure 52 where we gave the WIDL specification of a parameter table on the left-hand side with its visualization on the right-hand side.

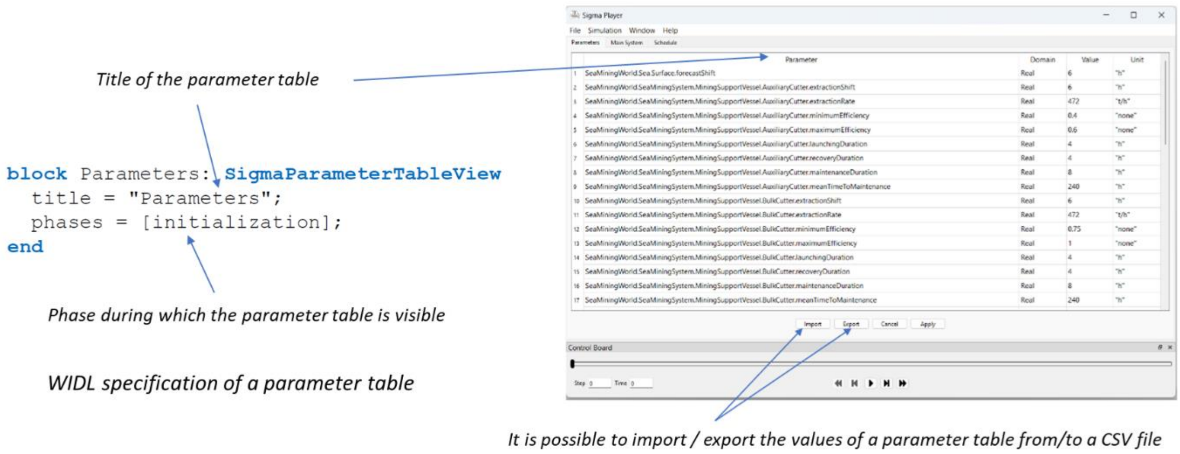


Figure 52 – Parameter table specified in WIDL with its corresponding visualization

- *View 2 – system list*: a **system list**, whose basic class is **SigmaSystemListView,** displays all elements of a ΣTM model in a 1-D tree view, as illustrated in Figure 53 where we gave the WIDL specification of a system list on the left-hand side with its visualization on the right-hand side.

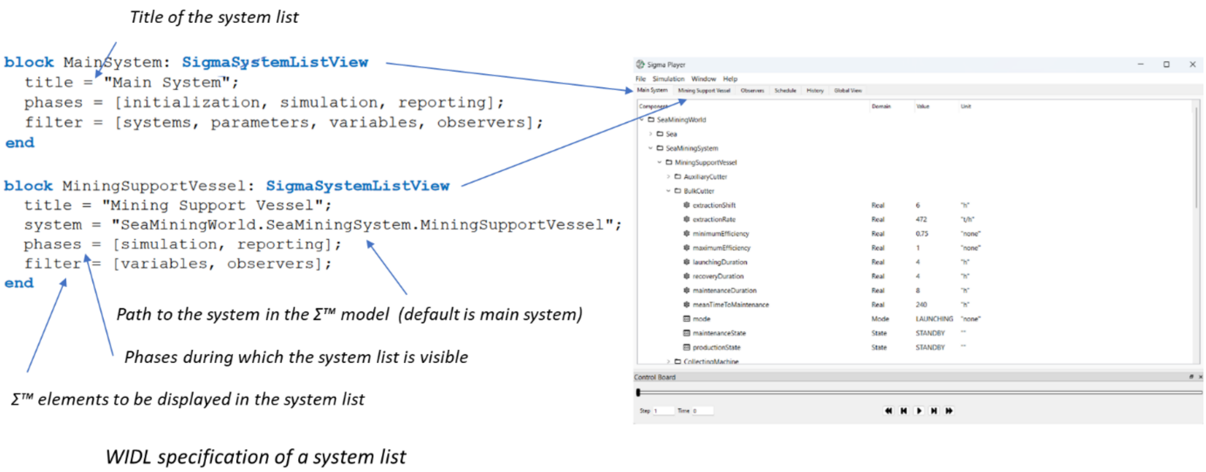


Figure 53 – System list specified in WIDL with its corresponding visualization

- *View 3 – observer table*: an **observer table**, whose basic class is **SigmaObserverTableView**, displays **all observers of a Σ™ model** with their associated standard statistics (value, sum, mean, minimum, maximum, first change time, number of changes), as illustrated in Figure 54 where we gave the WIDL specification of an observer table on the left-hand side with its visualization on the right-hand side.

Title of the observer table

```
block Observers: SigmaObserverTableView
  title = "Observers";
  phases = [simulation, reporting];
end
```

Phases during which the observer table is visible

WIDL specification of an observer table

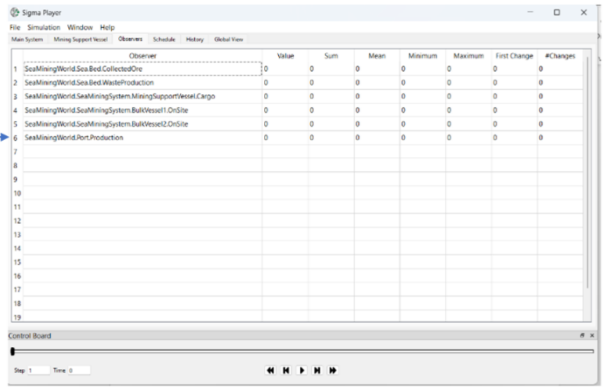


Figure 54 – Observer table specified in WIDL with its corresponding visualization

- *View 4 – schedule table*: a **schedule table**, whose basic class is **SigmaScheduleTableView**, displays the **next events to be fired** in a standard table containing the following data for each event (start, completion, observation): step, time, involved actor (full path), activity, event, as illustrated in Figure 55 where we gave the WIDL specification of a schedule table on the left-hand side with its visualization on the right-hand side.

Title of the schedule table

```
block Schedule: SigmaScheduleTableView
  title = "Schedule";
  phases = [initialization, simulation, reporting];
end
```

WIDL specification of a schedule table

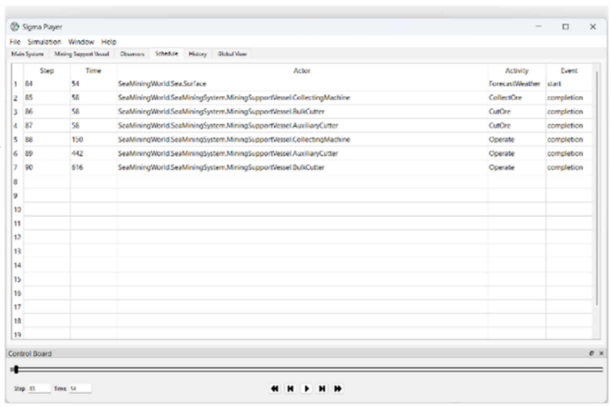


Figure 55 – Schedule table specified in WIDL with its corresponding visualization

- *View 5 – history table*: an **history table**, whose basic class is **SigmaHistoryTableView**, displays the **n last events that have been fired** during a given simulation of a Σ™ model (the number n being a parameter of the simulation) in a standard table containing the same data than in a schedule table., as illustrated in Figure 56 where we gave the WIDL specification of a history table on the left-hand side with its visualization on the right-hand side.

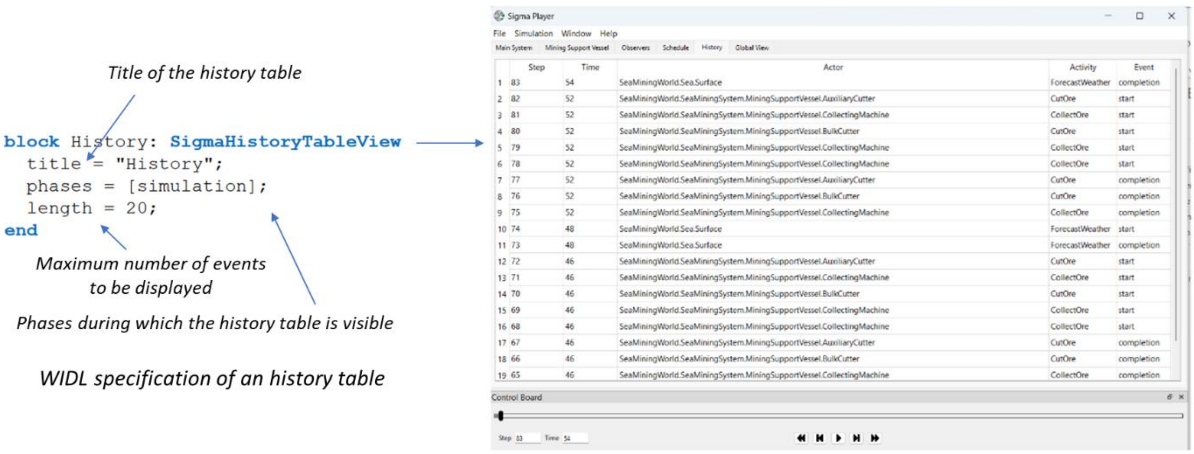


Figure 56 – History table specified in WIDL with its corresponding visualization

- *View 6 – diagram***:** a **diagram**, whose basic class is **Diagram**, allows to implement ad hoc views. A diagram is basically a scene where one can position graphical objects such as rectangles, ellipses, lines, texts and images, as illustrated in Figure 57.

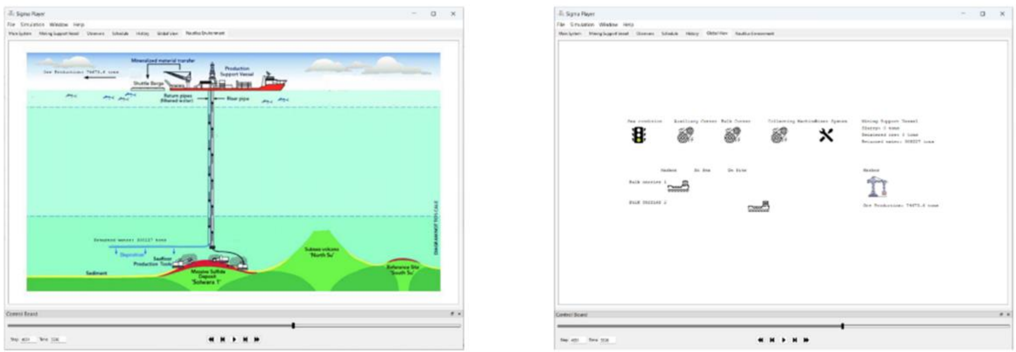


Figure 57 – Examples of diagram visualizations following a specification in WIDL

A typical example of diagram specification in WIDL is provided in Figure 58.

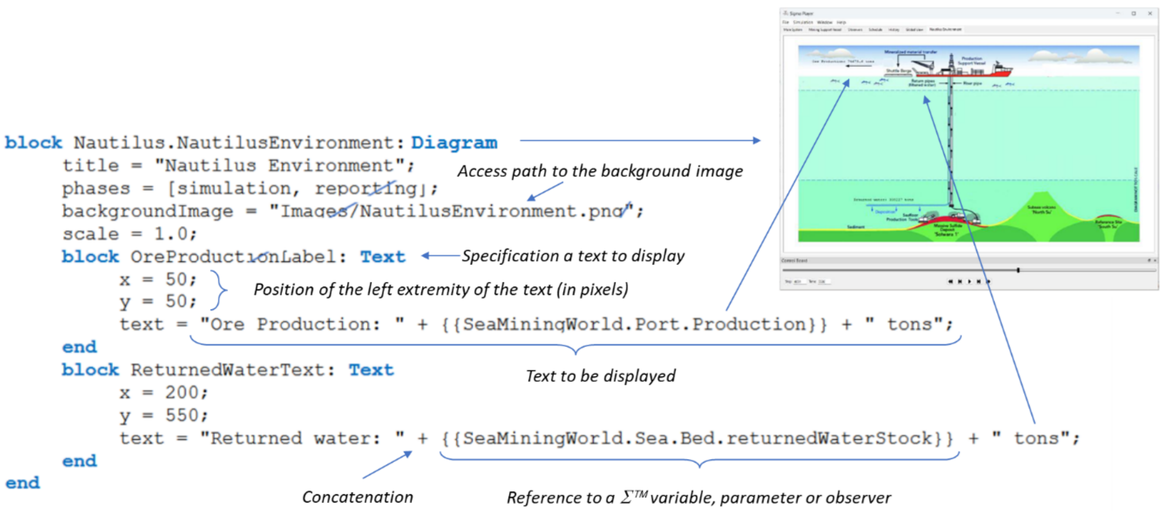


Figure 58 – Diagram specified in WIDL with its corresponding visualization

Rectangles, ellipses, lines and texts can be especially specified within a **diagram**, using the specification syntax which is illustrated in Figure 59.

```
block MyRectangle: Rectangle
    x = 140;
    y = 10;
    width = 200;
    height = 100;
    foregroundWidth = 2;
    foregroundColor = "black";
    backgroundColor = "white";
end
```

x, y: Position of the top-left corner (in pixels)
width, height: Width & height of the rectangle

```
block MyLine: Line
    x1 = 140;
    y1 = 10;
    x2 = 200;
    y2 = 100;
    foregroundWidth = 2;
    foregroundColor = "red";
end
```

x1, y1: Position of the first extremity of the line
x2, y2: Position of the last extremity of the line
foregroundWidth: Thickness of the line
foregroundColor: Color of the line

```
block MyEllipse: Ellipse
    x = 140;
    y = 10;
    width = 200;
    height = 100;
    foregroundWidth = 2;
    foregroundColor = "black";
    backgroundColor = "white";
end
```

x, y: Position of the top-left corner of the bounding rectangle (in pixels)
width, height: Width & height of the bounding rectangle
foregroundWidth: Thickness of the frame (in pixels)
foregroundColor, backgroundColor: Colors of the frame and of the background

```
block MyText: Text
    x = 200;
    y = 550;
    fontStyle = "normal",
    fontSize = 10,
    fontWeight = "normal",
    fontFamily = "Courier",
    foregroundColor = "black",
    text = "Returned water: " + {{SeaMiningWorld.Sea.Bed.returnedWaterStock}} + " tons";
end
```

fontStyle: Font style shall be in { normal, italic, oblique }
fontFamily: Font family depends on the operating system

```
block MyLine: Line
    x1 = 140;
    y1 = 10;
    x2 = 200;
    y2 = 100;
    foregroundWidth = 2;
    foregroundColor = "red";
    foregroundStyle = "dash";
end
```

foregroundColor: Color can be either a predefined color within { white, red, green, blue, black, darkRed, darkGreen, darkBlue, cyan, magenta, yellow, grey, darkCyan, darkMagenta, darkYellow, darkGrey, lightGrey } or any RGB color (e.g. #0ACC99)
foregroundStyle: Line style can be any value in { solid, dash, dot, dashDot, dashDotDot }

Figure 59 – Specifications of a rectangle, a line, an ellipse and text within a diagram in WIDL

Finally **images** can also be specified within a **diagram**, with the syntax illustrated in Figure 60.

```
block Icon: Image
    x = 10;
    y = 25;
    mode = {{SeaMiningWorld.SeaMiningSystem.MiningSupportVessel.AuxiliaryCutter.mode}};
    file = if mode==OPERATION
           then "Images/MachineInOperation.png"
           else "Images/MachineInMaintenance.png";
end
```

x, y: Position of the top left corner of the image (in pixels)
file: Position of the file containing the image (relative to the project folder path)

Figure 60 – Specification of an image within a diagram in WIDL

- *View 7 – group*: one can gather several elements in a **single logical structure** which is called a **group**, as illustrated in Figure 61. The graphical positions of the elements of the group are then relative to the top left corner of the group.

```
block MyGroup: Group
        x = 10;  }
        y = 25;  }  Position of the top left corner of the group (in pixels)
        block SubBlock1 … end
        block SubBlock2 … end
        …
end
```

Figure 61 – Specification of a group in WIDL

- *View 8 – block diagram*: WIDL provides a certain number of graphical elements that ease the construction of **block diagrams**. Diagram blocks can be indeed seen as the combination of a rectangle and of a group. Their generic structure is illustrated in Figure 62?

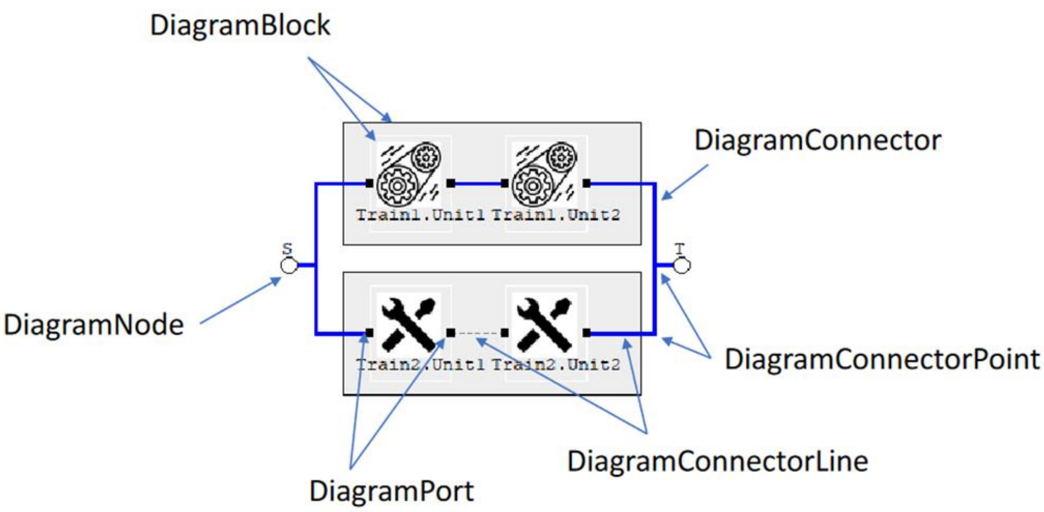


Figure 62 – Generic structure of a block diagram in WIDL

Ports are small black rectangles located at the border of a block diagram. **Diagram nodes** are small white circles. **Connectors** are groups of **connector points** (which are normally invisible) and **connector lines** (whose extremities are either connector points within the same connector, either ports of blocks or diagram nodes). The width, color and style of a connector is set at connector level and applies to all its components.

In these matters, Figure 63 provides then a part of the WIDL specification of the block diagram whose visualization is given by Figure 62.

.

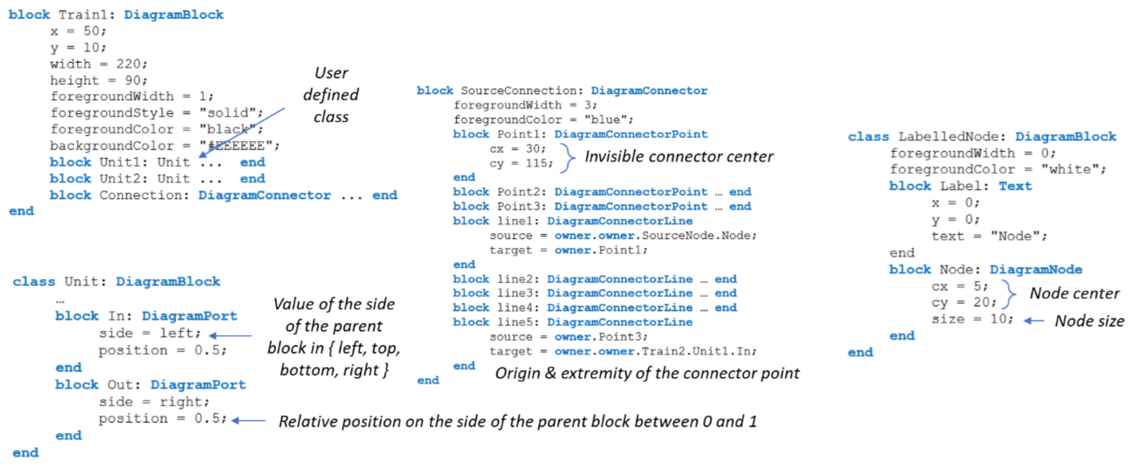


Figure 63 – Specification of a block diagram in WIDL with its visualization

One can then use these different view constructions to specify user interfaces in WIDL that can then be visualized through the compilation of the Σ™ model of a given system. An example of a quite simple user interface using this mechanism is provided in the below Figure 64.

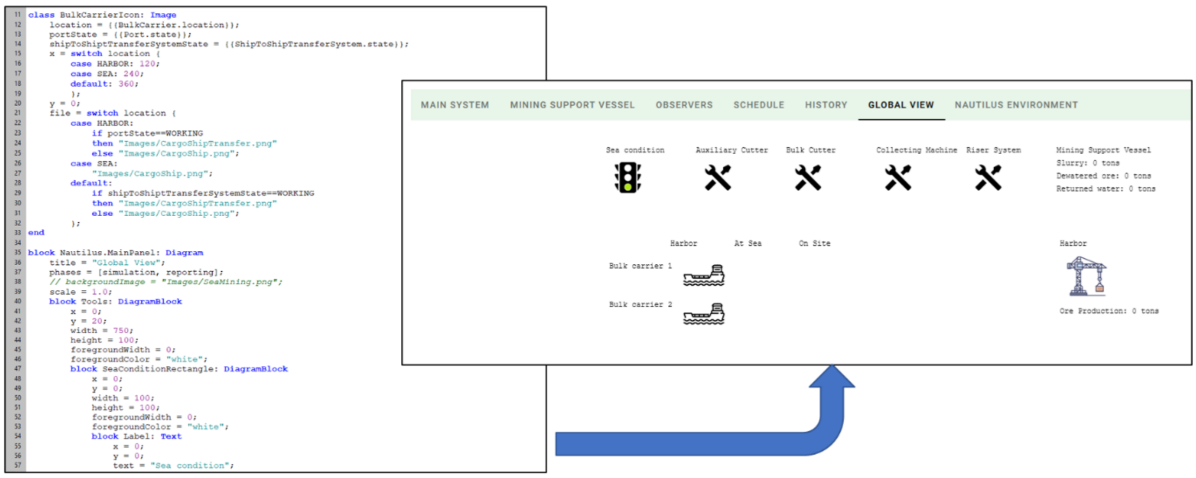


Figure 64 – Specification of an user interface in WIDL with its visualization