



Elements of System Dynamics

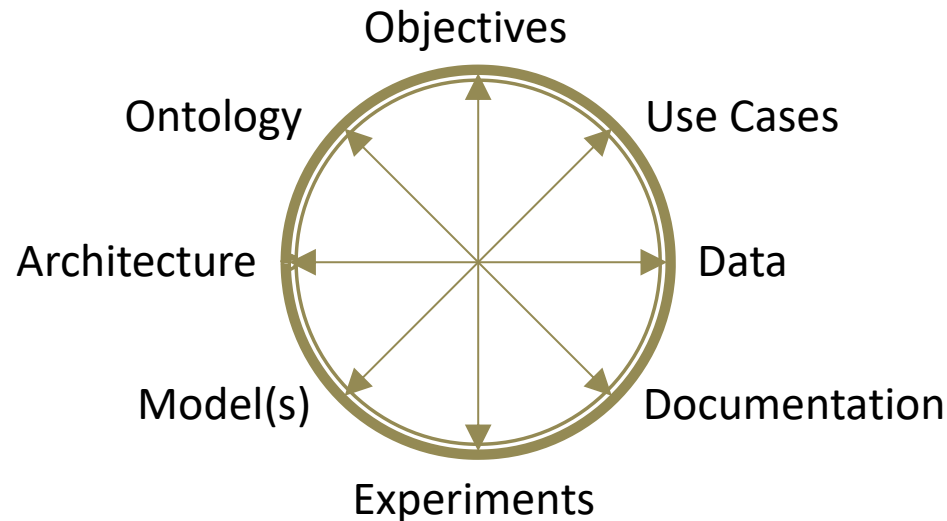
Antoine B. Rauzy

PART 4. SYSTEMIC DIGITAL TWINS

LECTURE 7. THE Σ^{TM} MODELING FRAMEWORK

Σ^{TM} Modeling Framework

The Σ^{TM} **modeling framework** helps to design systemic digital twins efficiently and accurately. It consists of **activities** to be performed resulting in as many **artifacts**:



The analyst **must adapt** the Σ^{TM} modeling framework to each specific study. It is a general **guideline**, not a strict procedure to follow.

LECTURE 7. THE Σ^{TM} MODELING FRAMEWORK

Key notions:

- V-cycle, Iterative Design
- Pragmatic Proofs
- Key Performance Indicator
- Requirement
- Ontology
- Use Cases
- BPMN
- Logical and Process Architecture
- Stocks and Flows
- Influence Diagrams

Agenda

Section 1. Preliminary Remarks

Section 2. Case Study

Section 3. Objectives

Section 4. Ontology

Section 5. Use Cases

Section 6. BPMN

Section 7. Dynamics

Section 8. Data

Section 9. Models

Section 10. Documentation

Section 11. Experiments

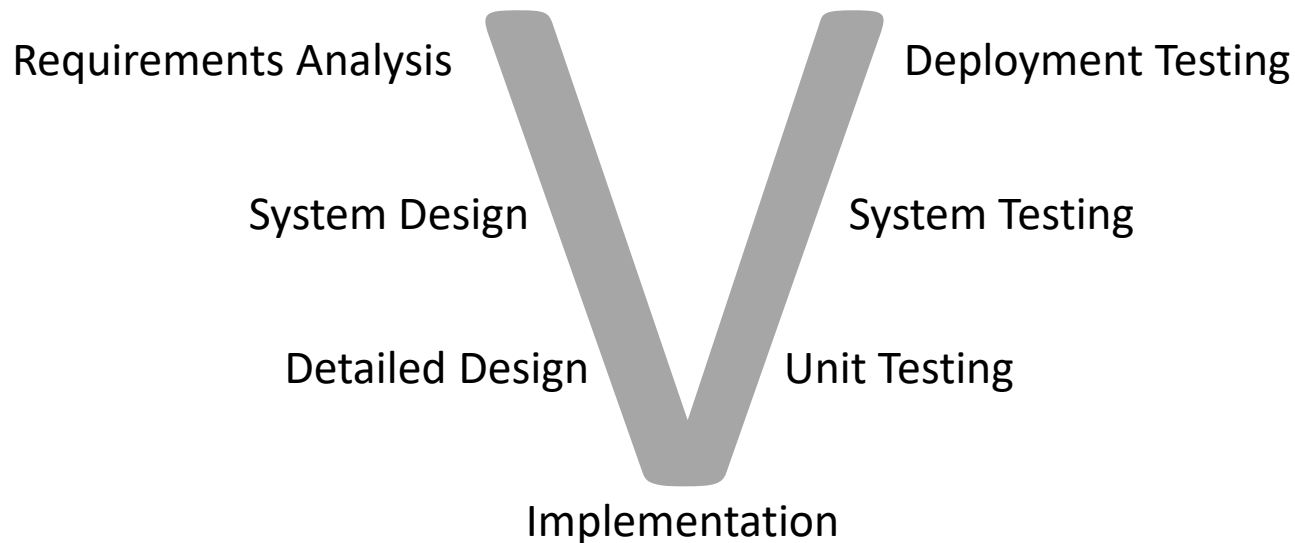
Section 12. Wrap-Up

LECTURE 7. THE Σ^{TM} MODELING FRAMEWORK

SECTION 1. PRELIMINARY REMARKS

The Mythical V-Cycle

The **V-cycle** is a model used primarily in **systems engineering** and **software development** to represent the process of planning, developing, and validating complex systems. The V-shape represents two key phases: **development** on the left side of the "V" and **testing/validation** on the right side.



The V-cycle emphasizes **sequential development** with rigorous testing.

Iterative Design

The V-cycle (and similar methodological frameworks) assumes implicitly that the system of interest is designed from scratch, which is **never** the case in practice.

Systems are essentially designed/built by assembling **existing components**, with a different degrees of maturity. These components come with their specifications and tests. The design process consists then in **maturing** and **assembling** these components, **making decisions all the way**.

The V-cycle is **logical** and not chronological.

Likewise, the design a **systemic digital twin** is an **iterative process**:

- Reuse modeling components, patterns and metaphors;
- Never try to design "the" final model directly, do not hesitate to design intermediate models and even side models.
- Test your models each time you add them a new feature.

Models as Proofs

Models (and the associated experiments *in silico*) are **working tools**, not (platonic) ideals the system should comply with.

A **systemic digital twin** should be seen as a **constructive** and **pragmatic proof** that it is possible to design, to build and to operate a system having some specified properties.

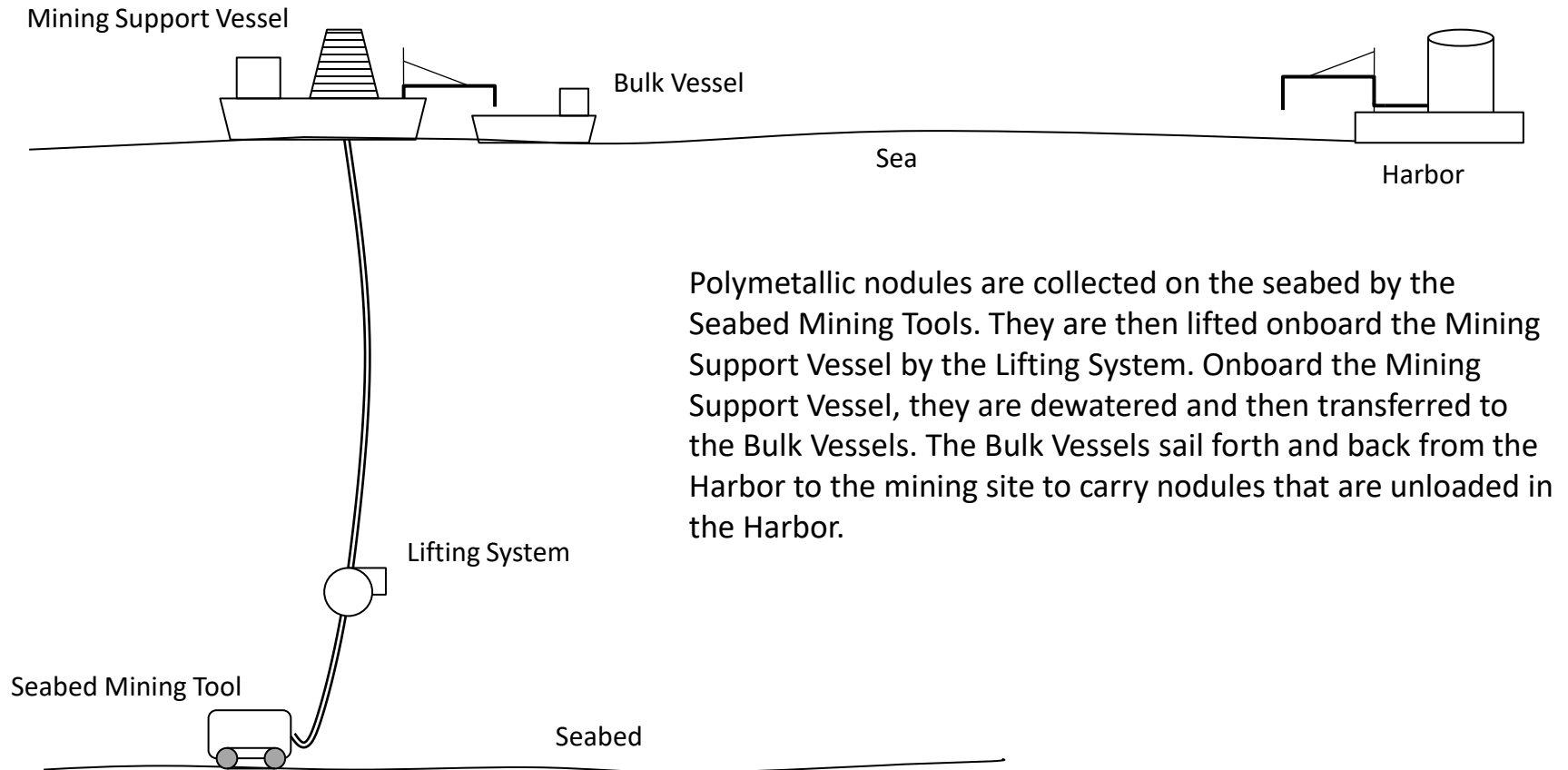
Constructive proofs: One shows the existence of an object by constructing a concrete example of this object (no *reductio ad absurdum*)

Pragmatic proofs: Conversely to mathematical proofs, proofs in systems engineering involve assumptions on the physical world. Proofs are thus conditional to these assumptions.

LECTURE 4. THE Σ^{TM} MODELING FRAMEWORK

SECTION 3. CASE STUDY

Sea Mining System

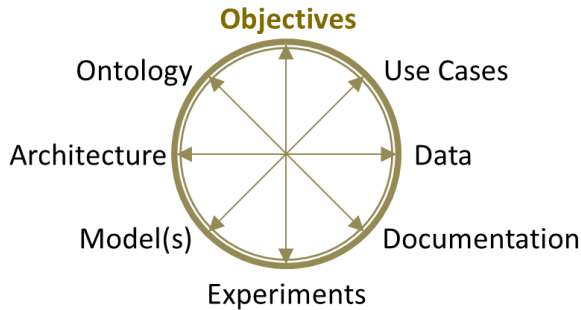


[Solheim 2023] Assessment of expected production of a deep-sea mining system: An integrated model-based systems engineering and discrete event simulation approach Astrid V. Solheim, Antoine B. Rauzy, Per Olaf Brett, Steinar Ellefmo, Tonje Hatling, Rudy Helmons, and Bjørn Egil Asbjørnslett In *Systems Engineering*. Wiley. 2023. doi: 10.1002/sys.21699

LECTURE 4. THE Σ^{TM} MODELING FRAMEWORK

SECTION 2. OBJECTIVES

Objectives



Developing a systemic digital twin is a resource consuming, committing process. If one develops a systemic digital twin, one must have good reasons to do so. It is thus of primary importance to **state**, with as few ambiguities as possible, what are the **objectives of the study**.

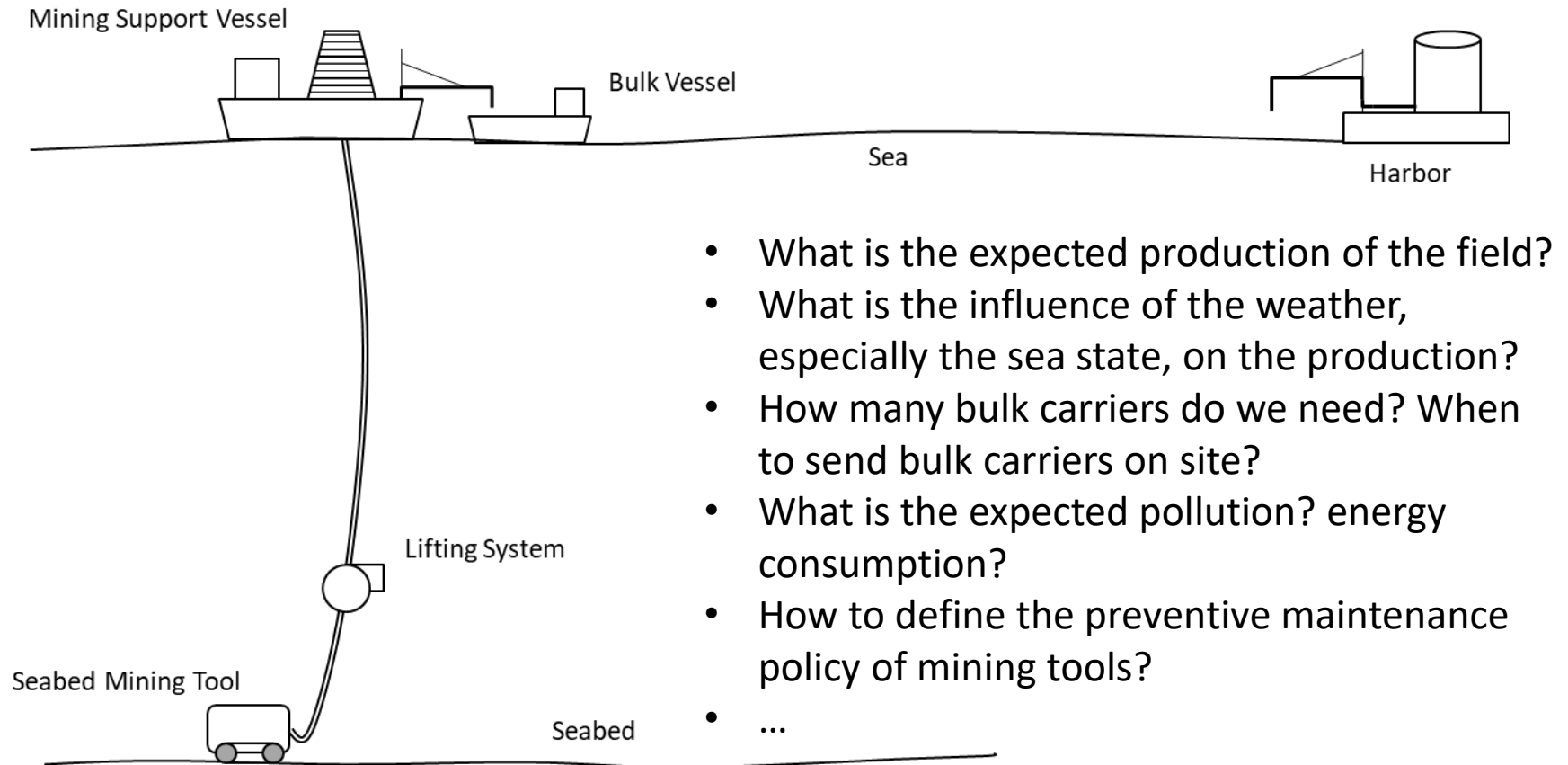
A systemic digital twin is developed by an **analyst** for a **client** (who is not a specialist of system dynamics). The objectives of the study, as stated by the analyst, must be **understood** and **agreed** by the client.

Key questions:

- What is the **system** of interest?
- What are the **issues** to be looked at?
- *What do you want to **learn** about the system?*

It is often a good to design small, **metaphorical models**, to illustrate the objectives.

What Do We Want to Learn?



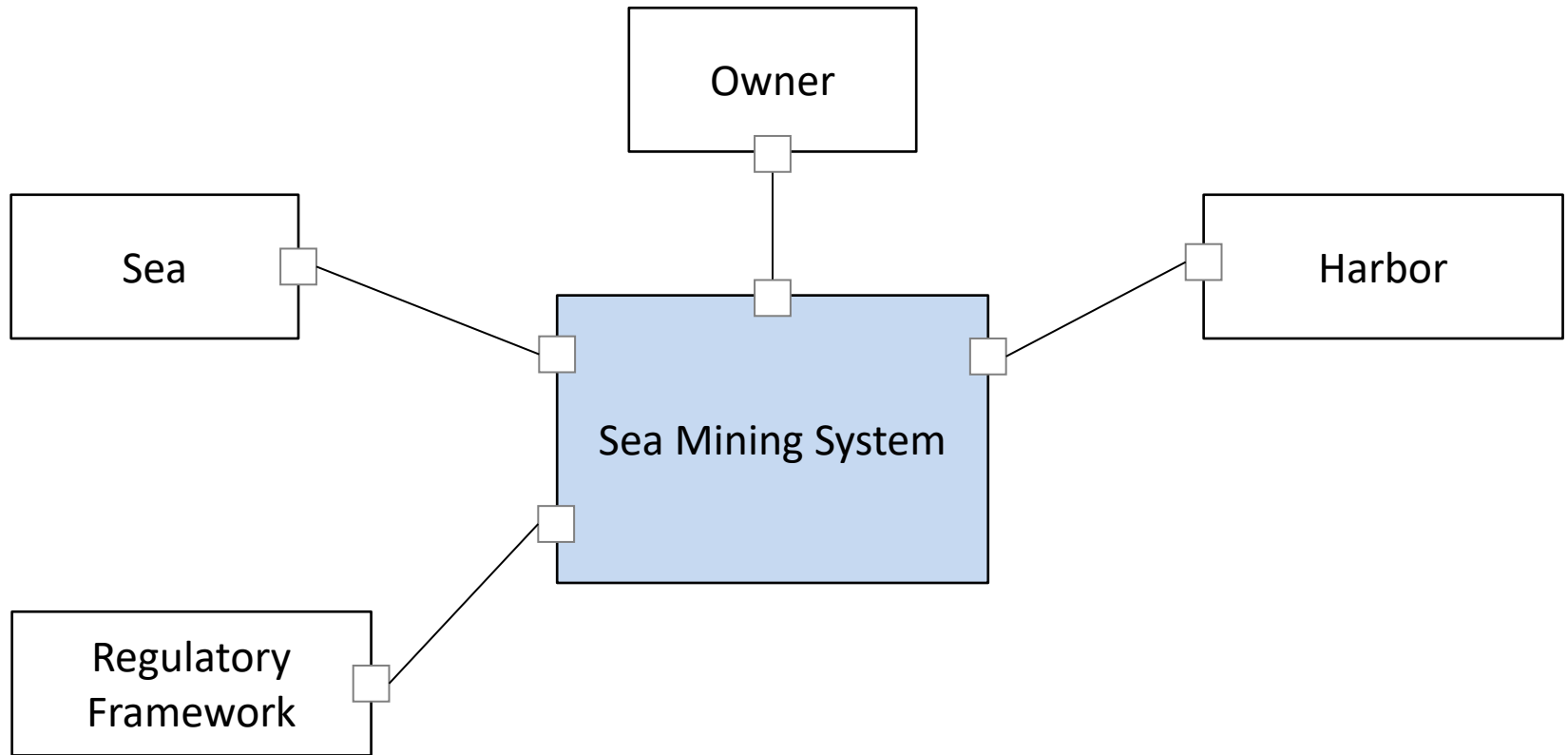
Requirements

It is of key importance to turn informal questions into less ambiguous statements. Requirements are of great help to do so.

Name	Reference	Maturity
Safety	REQ 001	Weak
Statement		
The expected production of the sea mining system should be over 10^6 tons of ore per year.		
Justification		
The sea mining operation are profitable only if this threshold is reached		
Satisfaction criterion		
Design a systemic digital twin of the sea mining system and assess they yearly annual production from that model.		

Environment Analysis

Defining the boundaries of the system under study and its key interaction with its environment is very important to better understand what is at stake, although the systemic digital twin involves necessarily a description of the system and its environment.



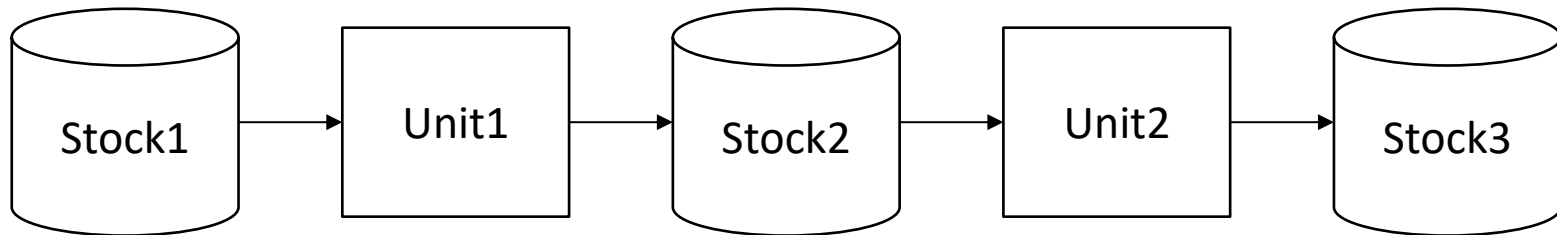
Interactions

Interactions between the system and external systems must be characterized.
Only **direct interactions** (and the corresponding external systems) should be considered.

External System	Interaction
Owner	Expected production
Sea	Density of minerals on the seabed. Sea condition on the surface.
Regulatory framework	Pollution, safety
Harbor	Siloes capacity

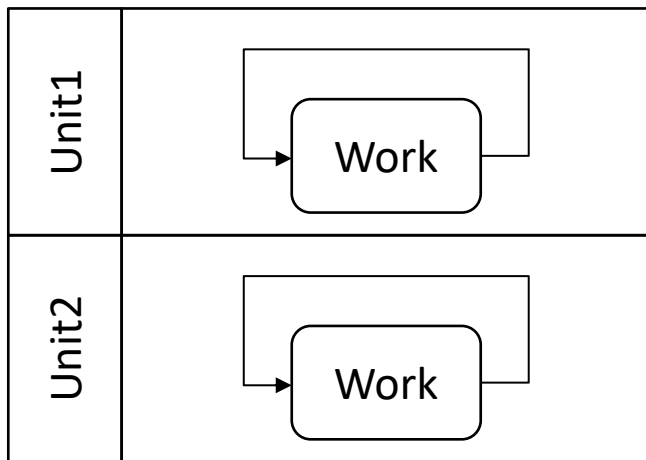
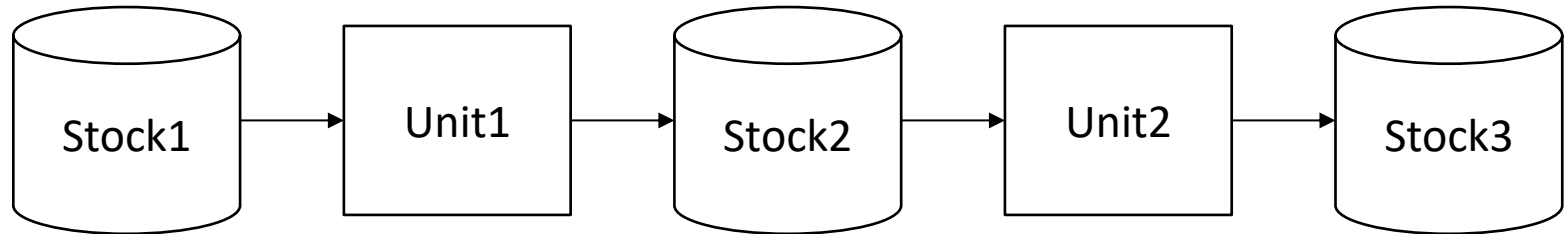
Key Metaphor: Supply Chain (1)

In linguistic, a **metaphor** is a figure of speech that describes something by referring to something else with similar qualities. In the case of models, a metaphor is a model, or a modeling principle, that is easy to understand even by non-specialists and that summarizes the objectives of the model.



Actor	Activity	Trigger	Start	Completion	Duration
Unit1	Work	not working $\text{Stock1} \geq \text{quantity1}$	working = true $\text{Stock1} -= \text{quantity1}$	working = false $\text{Stock2} += \text{quantity2}$	duration1
Unit2	Work	not working $\text{Stock2} \geq \text{quantity3}$	working = true $\text{Stock2} -= \text{quantity3}$	working = false $\text{Stock3} += \text{quantity4}$	duration2

Key Metaphor: Supply Chain (2)



Units of the supply chain work in parallel, at different paces and treat different quantities. Both are subordinated to external conditions and may be subject to random variations.

A particular attention should therefore be put on the possible bottlenecks of the process and consequently of the treatment strategy.

Importance of Defining Objectives

Once more: one does not develop a systemic digital twin to mimic the "reality" but to learn about the system under study, i.e. to **answer concrete questions**.

The systemic digital twin must be the most **parsimonious** to achieve this goal.

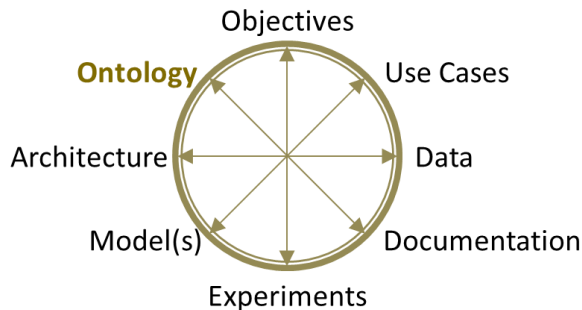
Defining clearly the objectives help:

- To define the right **level of abstraction** of the model.
- To define the right **unities of the variables** of the model.
- To define the right **time step** of the model.

LECTURE 4. THE Σ^{TM} MODELING FRAMEWORK

SECTION 4. ONTOLOGY

Ontology



“In information science, an **ontology** encompasses a representation, formal naming, and definitions of the categories, properties, and relations between the concepts, data, or entities that pertain to one, many, or all domains of discourse” [Wikipedia].

To a design of a systemic digital twin, an ontology must at least contain:

- A **glossary** defining of all **technical terms** used in the domain of the system of interest.
- A **conceptual data model** including:
 - A description of core **entities** of the system and its environment;
 - A description of **relationships** existing between these entities;
 - A description of **business rules and constraints** that apply on entities and their relationships

Foundational Ontologies

In information science, a **foundational ontology** is an ontology that consists of very general terms (such as "object", "property", "relation") that are common across all domains, i.e. an ontology of ontologies. Foundational ontologies are also called upper ontologies, or top-level ontologies.

BFO and DOLCE are attempts to provide such foundational ontologies. However, despite of years of efforts, no unified framework emerged so far, and specialists are pessimistic about the possibility of such framework.

Important point: Ontologies, at least in the way we use them to design systemic digital twins, are **pragmatic descriptions** of the universe of discourse. Their objective is to clarify the concepts at work, not to make them fully formal (mathematized).

Σ^{TM} Ontology

Systems	• Fundamental entities we are looking at • Containers for systems, variables and activities
Variables	• Attributes/properties of systems • Value holders (integer, float, Boolean...) • Define states of systems
Activities	• Mechanisms by which the state of systems evolve / describe the dynamics of systems • Are (implicitly) organized into processes

Cloning	• Similarity of systems at model level
Classes/Instances	• Similarity of systems at domain level

Composition	• <i>is-part-of</i> relation
Inheritance	• <i>is-a</i> relation (extends directive)
Aggregation	• <i>uses</i> relation (ports, virtual variables).

What is Missing?

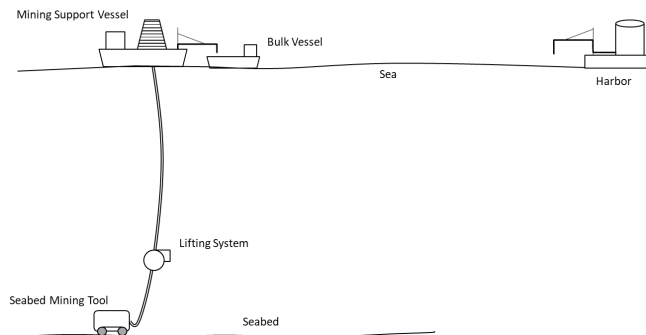
The Σ^{TM} ontology is not expressive enough to be foundational.

- The keyword *system* may be a bit misleading, expect if we accept to call system any entity of interest.
- Need for a way to describe:
 - **Abstract/parametric systems:**
 - + A concrete sea mining system composes a definite number of bulk carriers.
 - + An abstract sea mining system composes an indefinite number of bulk carriers.
 - **Deformable systems:**
 - + Systems whose architecture varies through their mission, systems of systems, e.g. battlefields.
 - **Business rules and constraints:**
 - + Minimum and maximum values of a parameter
 - + Minimum and maximum number of bulk carriers
- Need for a more explicit way of describing **processes**.
 - See sections on Use Cases and Dynamics

Sea Mining System Ontology (1)

system: SeaMiningSystem

description: A sea mining system extracts minerals (like copper, nickel, cobalt) from the deep seabed using large, remotely operated vehicles (ROVs) that collect polymetallic nodules or break apart crusts/sulfides, pump the slurry/ore up to a surface vessel via a riser, separate the minerals, and return sediment/water to the ocean.



composing systems: SeaMiningWorld

sibling systems: Sea, Harbor

composed systems: MiningSupportVessel, BulkCarrier [1, 5]

minimum and maximum numbers

Sea Mining System Ontology (2)

class: BulkCarrier

description: A bulk carrier is a large merchant ship designed to transport unpackaged dry cargo, like coal, grain, iron ore, cement, and fertilizers, in its large, unobstructed cargo holds, facilitating global trade in essential raw materials. Key features include a flat deck with large hatch covers for easy loading/unloading, and they are vital for moving commodities efficiently, often using cranes or conveyor systems, with sizes varying greatly.



composing systems: SeaMiningSystem

sibling systems: MiningSupportVessel

characteristics:

- storageCapacity(type: float, unit: tons, default: 25000)
- cruiseSpeed (type: float, unit: NM/h, minimum: 12, maximum: 14)

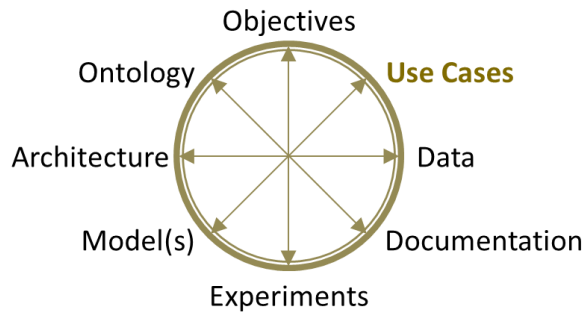
Recommendations

- Present ontologies as hypertexts
 - Use e.g. Tau to manage references (and styles)
 - Do not hesitate to illustrate descriptions with pictures and diagrams
- Ensure that all key concepts of the domain of interest are defined.
 - At least all systems of the model should have an entry in the ontology
- Ensure that all concepts mentioned in the ontology are defined
- When defining characteristics, always think what are their units
 - Ensure that your unit system is consistent
 - Give typical/default values and when applicable minimum and maximum values

LECTURE 4. THE Σ^{TM} MODELING FRAMEWORK

SECTION 5. USE CASES

Use Cases



"In both software and systems engineering, a **use case** is a structured description of a system's behavior as it responds to requests from external actors, aiming to achieve a specific goal" [Wikipedia]

External systems, whether they are human beings or not, are often called **actors**.

Use cases have three main roles:

1. They avoid **abstract non-sense** by making things very concrete.
2. They cover the main **scenarios of use** of the system.
3. They prepare **tests** to be performed on the system, once built.

In the case of systemic digital twins, the analyst may design actually two series of use cases:

- Use cases describing the **behavior** of the **system** of interest, and
- Use cases describing the **scenarios** of use of the **digital twin** itself.

Definition and Role

In software and systems engineering, a **use case** is a sequence of actions or steps defining the interactions between the system under study, or parts of it, and one or more external systems.

External systems, whether they are human beings or not, are often called **actors**.

As a rule, for each feature introduced somewhere in one of the model(s), there must be at least one use-case involving this feature.

Operating the Mining Tool (plain text)

Precondition: The mining tool is ready to be operated.

Postcondition: The collected ore is transferred to the lifting system.

Trigger: The mining tool is ready to operate and is not already operating.

Scenario:

1. The mining tool is positioned in the zone D5 of the seabed.
2. The mining tool collects the ore in this zone.
3. The amount of the collected ore depends on:
 - The density of ore in the zone D5.
 - The efficiency of the mining tool to collect ore.
4. The collected ore is transferred the lifting system.
5. The collection on a zone lasts 6 hours.

Operating the Mining Tool (markup language)

```
#+useCase OperatingMiningTool
#precondition The *MiningTool* is ready to be operated.
#postcondition The +collectedOre+ is transferred to the *LiftingSystem*.
#trigger The *MiningTool* is ready to operate and is not already
operating.
#+scenario
#action The *MiningTool* is positioned in the zone D5 of the *Seabed*.
#action The *MiningTool* collects the ore in this zone.
#remark The amount of the +collectedOre+ depends on:
- The +oreDensity+ in the zone D5.
- The +efficiency+ of the *MiningTool* to collect ore.
#action The +collectedOre+ is transferred the *LiftingSystem*.
#assert The collection on a zone lasts 6 hours.
#-scenario
#-useCase
```



*parameter
port
variable*



system

Operating the Lifting System (plain text)

Precondition: The lifting system is ready to be operated and there are some ore to lift-up.

Postcondition: The ore is transferred to the support vessel.

Trigger: The lifting tool is ready to operated and is not already operating.

Story:

1. The ore collected by mining tool is transferred to the support vessel.
2. The maximum lifting rate of lifted ore is 450 tons per hour.
3. The lifting shift lasts 6 hours.

Operating the Lifting System (markup language)

```
#+useCase OperatingLiftingSystem
  #precondition The *LiftingSystem* is ready to be operated and there is
    some +collectedOre+ to lift-up.
  #postcondition The +collectedOre+ is transferred to the *SupportVessel*.
  #trigger The *LiftingTool* is ready to operated and is not already
    operating.
  #+scenario
    #action The +collectedOre+ is transferred to the *SupportVessel*.
    #assert The +maximumLiftingRate+ of lifted ore is 450 tons per hour.
    #assert The +liftingShift+ lasts 6 hours.
  #+scenario
#-useCase
```

Operating the Dewatering System

Precondition: The dewatering system is ready to be operated and there are some ore to dewater.

Postcondition: The ore is dewatered.

Trigger: The dewatering tool is ready to operated and is not already operating.

Story:

1. The raw ore lifted on the support vessel is dewatered.
2. The quantity of dewatered ore is 10 000 tons per shift.
3. The dewatering process is not launch if there is not at least 10 000 tons of ore to dewater.
4. The dewatering shift lasts 24 hours.
5. By the dewatering process, the ore looses 80% of its weight.

Ship to Ship Transfer

Precondition: The bulk carrier must be on site

Postcondition: The bulk carrier is filled with dewatered ore

Trigger: The bulk carrier is on site.

Story:

1. The dewatered ore is transferred from the support vessel to the bulk carrier.
2. The bulk carrier is filled up to its capacity (28 000 tons).
3. The transfer takes place only if the weather forecast is favorable, i.e. if the state of the sea is good enough. Namely, there must be a time windows of at least 6 hours of favorable conditions for the transfer to be started.
4. The rate of transfer is about 500 tons per hour.

Recommendations

Writing use cases helps:

- To **elicit** actors, stocks, flows and activities.
- To **communicate** with non specialist stakeholders.

As a rule of thumb:

- There must be one use case for each key actor.
- There must be one use case for each key stock.
- There must be one use case for each key activity.

Use cases can be described textually with either **plain text** or with a **markup language**. They can also be represented graphically, e.g. as **BPMN** (see next section).

Recall that you do not need to write all use cases upfront. Moreover, you can make your use cases increasingly precise as you progress in the design of the systemic digital twin.

LECTURE 4. THE Σ^{TM} MODELING FRAMEWORK

SECTION 6. BPMN

Business Process Modeling and Notation

Business Process Model and Notation (BPMN) is a standard for business process modeling that provides a **graphical notation** for specifying business processes.

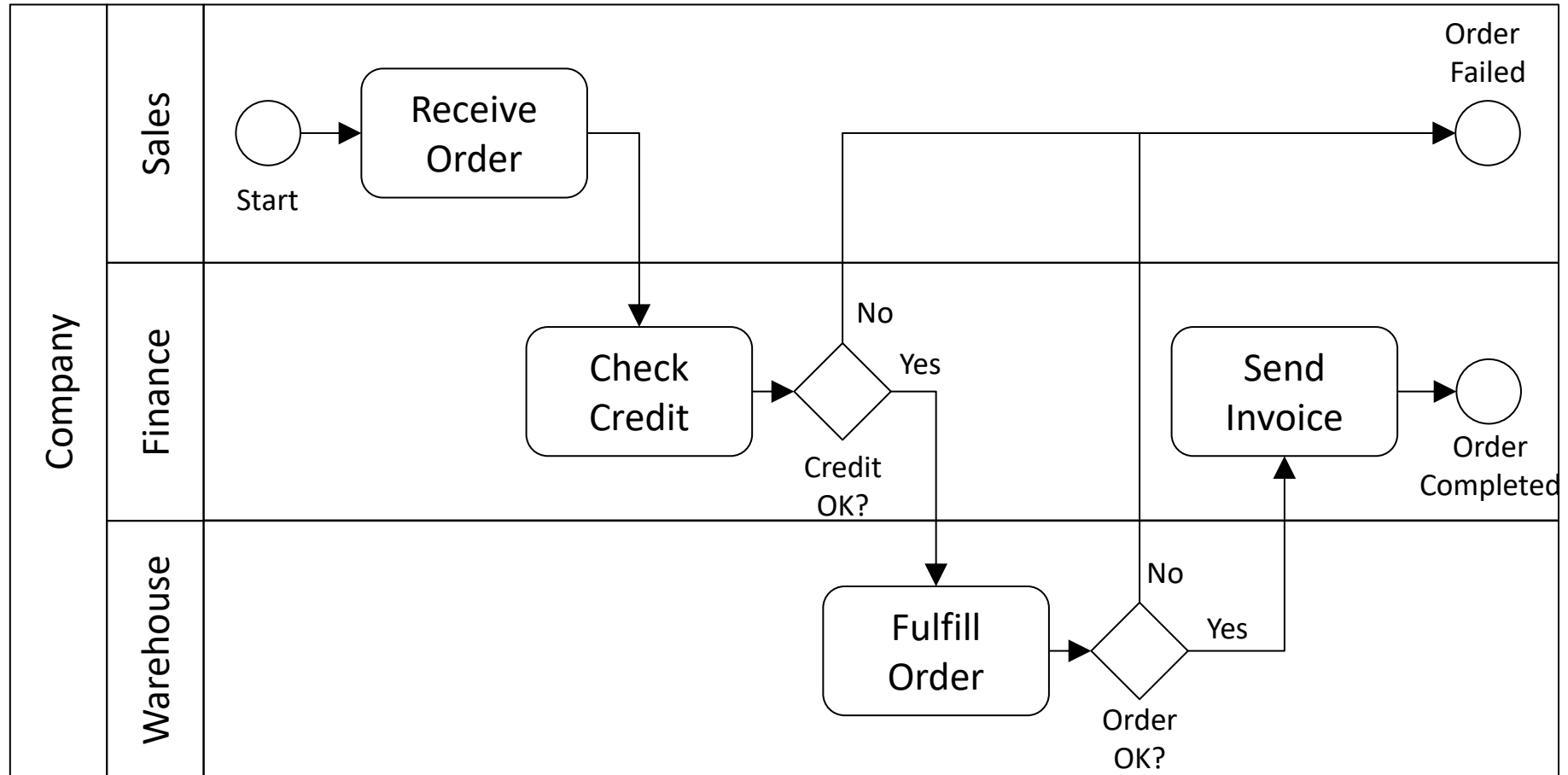
BPMN diagrams allow **stakeholders** to visualize business processes, making it easier to streamline **workflows**.

Originally developed by the Business Process Management Initiative (BPMI), BPMN has been maintained by the Object Management Group (OMG) since the two organizations merged in 2005. BPMN is also ratified as ISO 19510. The latest version is BPMN 2.0.2, published in January 2014.

BPMN's four basic element categories are:

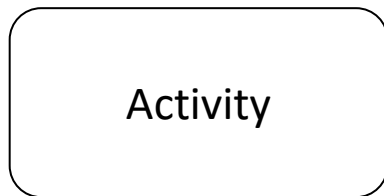
- Workflow elements: events, activities, gateways, sequence flows.
- Organizing elements: Pools, swim lanes, groups.
- Readability elements: links, annotations (we shall not use them).
- Special behavior elements: messages, signals, timers... (we shall not use them).

Processing of an Order

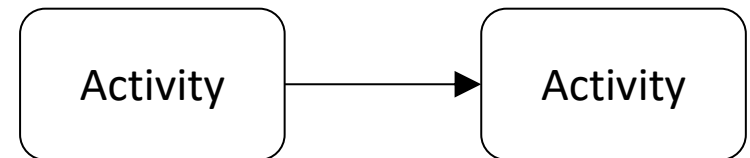


Workflow Elements

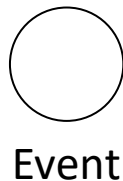
Activities are tasks that are performed in the process - by humans, by automation, or by subprocesses.



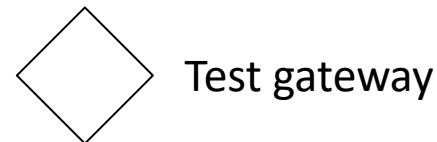
Sequences flow are used to show how the workflow moves.



Events are used to start or end a process, and to manage specific actions during a workflow.



Gateways are used to separate or join process flow.

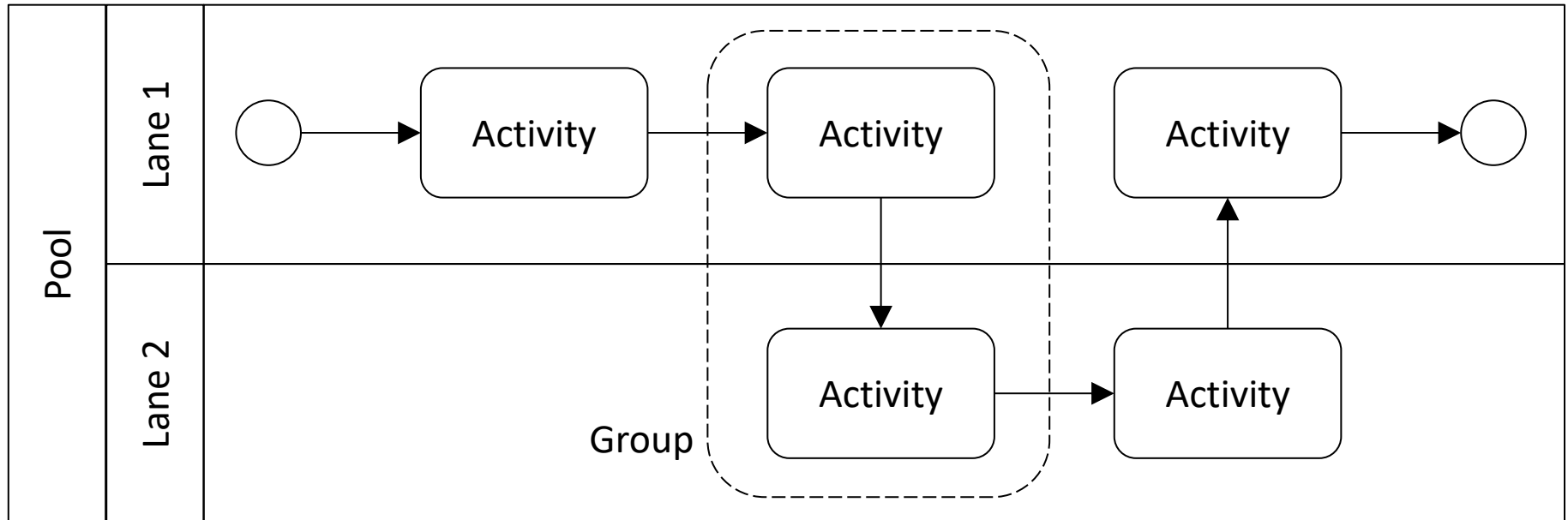


Organizing Elements

Pools contain a single, complete process. The workflow cannot leave its pool.

Swim lanes are used to tell who does what

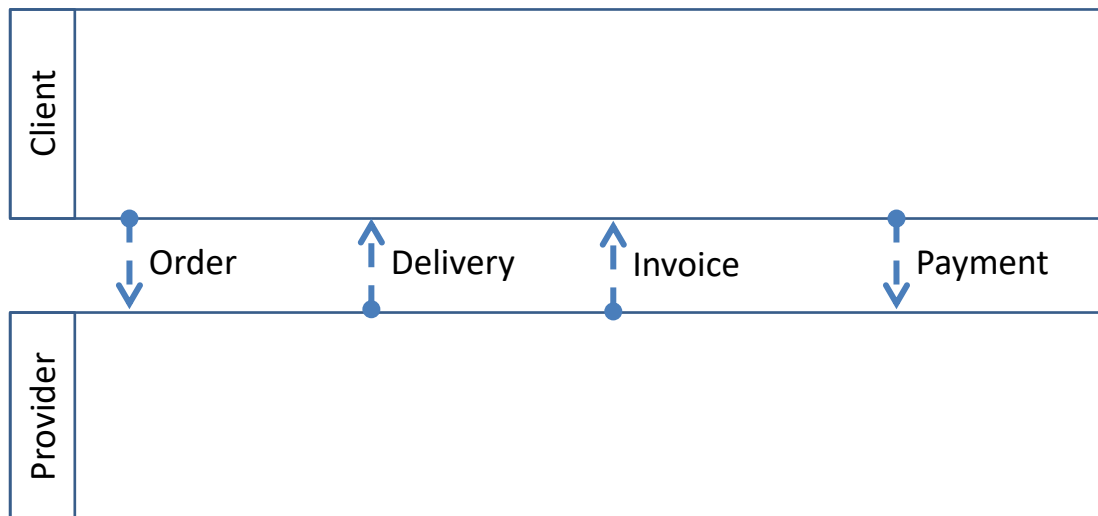
Groups are used for documentation purpose.



Orchestration and Choreography

Starting from version 2.0, BPMN distinguishes two types of processes:

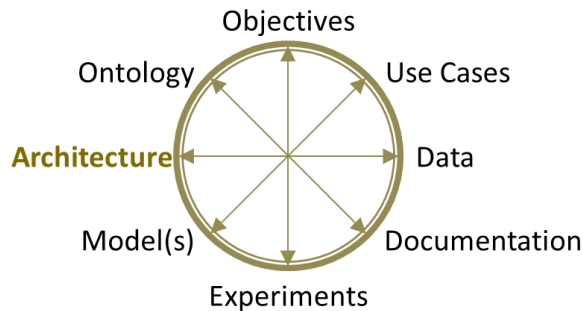
- **Orchestration**, which describes tasks performed by an actor or a group of actors sharing the same data/information (thus belonging to the same entity). The activities of different actors involved in an orchestration are represented in different lanes within a pool.
- **Choreography** which involves multiple entities communicating through message exchanges without sharing the same data/information (e.g., client/supplier). In this case, the activities of these different entities are represented in separate pools.



LECTURE 4. THE Σ^{TM} MODELING FRAMEWORK

SECTION 7. ARCHITECTURE

Architecture



Designing the ontology of the system of interest provides a preliminary view on this system. The question now is how this system is structured and how it evolves through the time? In other words, what is its **structure** of the system and what is its **behavior**?

A technical or socio-technical system can be seen as a hierarchical network of human and technical **actors**. This network is called the **logical architecture** of the system.

The global behavior of the system results from the **actions** of these actors. These actions results in general from **activities** themselves organized into **processes**. The system has thus also a **process architecture**.

The **stocks and flows** of matter, energy and information link the logical and the process architectures.

Studying the **dynamics** of a system means describing the actors, activities and processes at work in this system as well as its stock and flows.

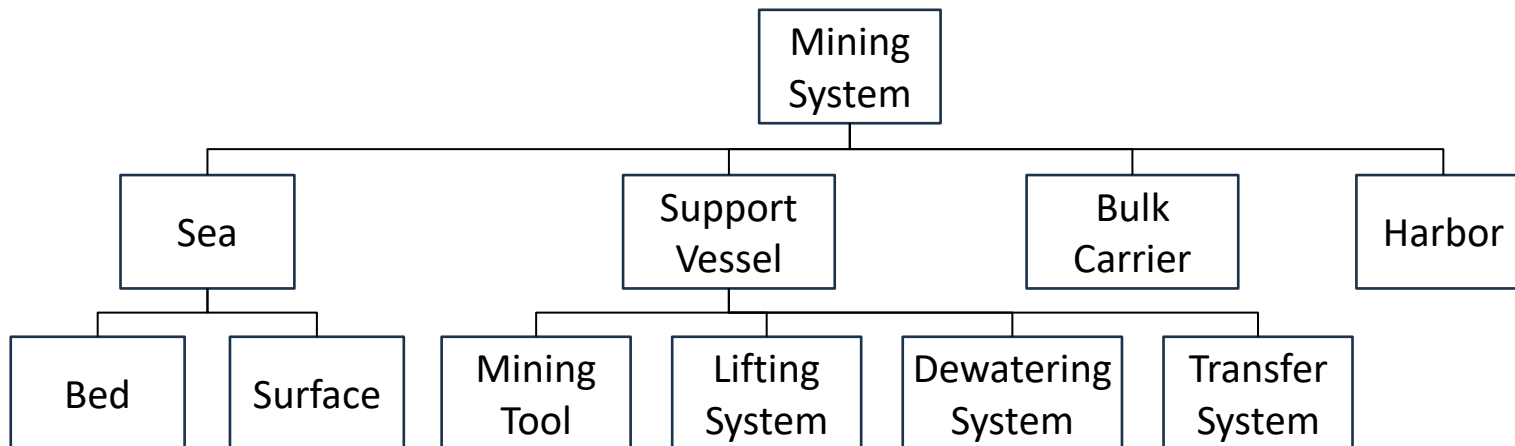
“System” must be understood here in a broad sense: some actors and activities to be considered may belong to the **environment** of the system. Similarly, the concept of activity includes not only planned activities, but also hazards, changes in the environment...

LECTURE 4. THE Σ^{TM} MODELING FRAMEWORK

SECTION 7.1. LOGICAL ARCHITECTURE

Logical Architecture

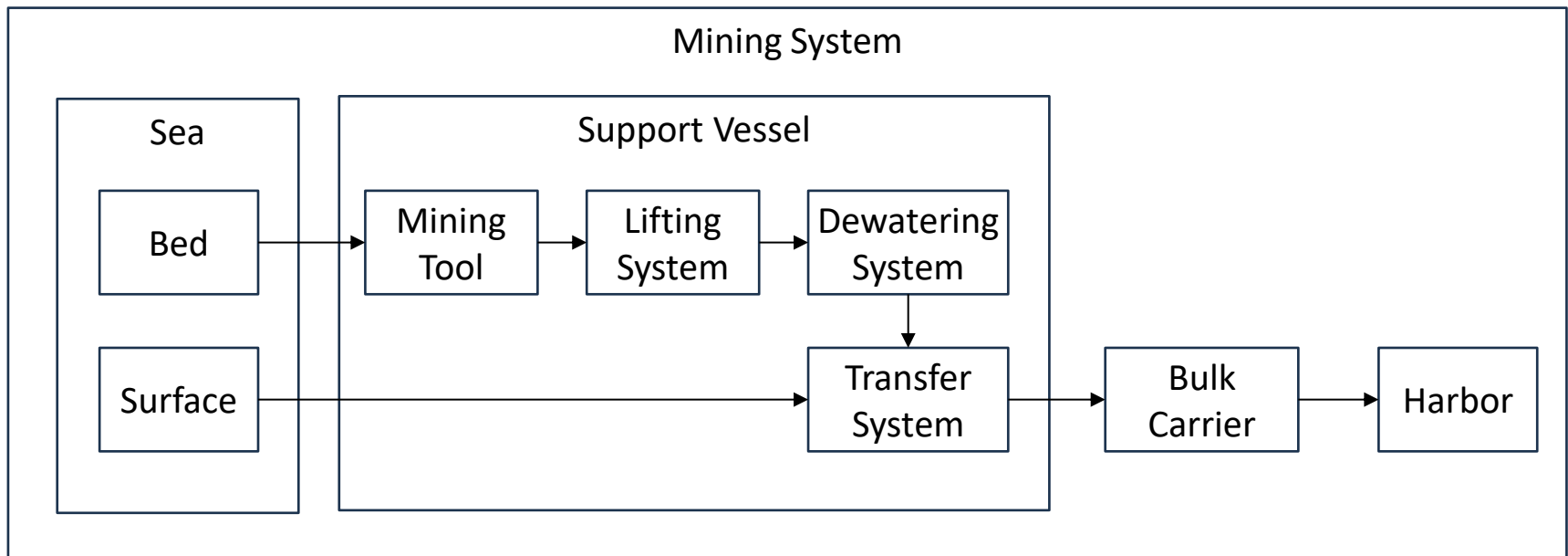
The **logical architecture** of the system is its **decomposition** into a **hierarchy of subsystems**. This breakdown aims at specifying and organizing the **actors** in the system as well as the **variables** (stocks) that characterize the state of the system.



Subsystems may be natural, human, hardware, or software. The decomposition is **logical** i.e. can involve both **physical** and **functional** elements. Several decompositions may be possible, depending on the point of view.

Block Diagrams

As an alternative to tree-like decompositions, **block diagrams** can be a suitable representation of the architecture of the system.



Block diagrams may be less appropriate than tree-like representations to understand the hierarchy of subsystems, but are better to represent **flows** of matters, energy and information circulating in the **network** of subsystems.

LECTURE 4. THE Σ^{TM} MODELING FRAMEWORK

SECTION 7.2. PROCESS ARCHITECTURE

Process Architecture

The objective of the **process architecture** is to determine the main **activities** taking place in the system and the **sequencing** of these activities.

It is tightly related with the **logical architecture** as activities are performed by **actors** and impact **subsystems**.

A good starting point consists in looking at the **life-cycle** of the system, i.e. its different **phases** or **modes**.

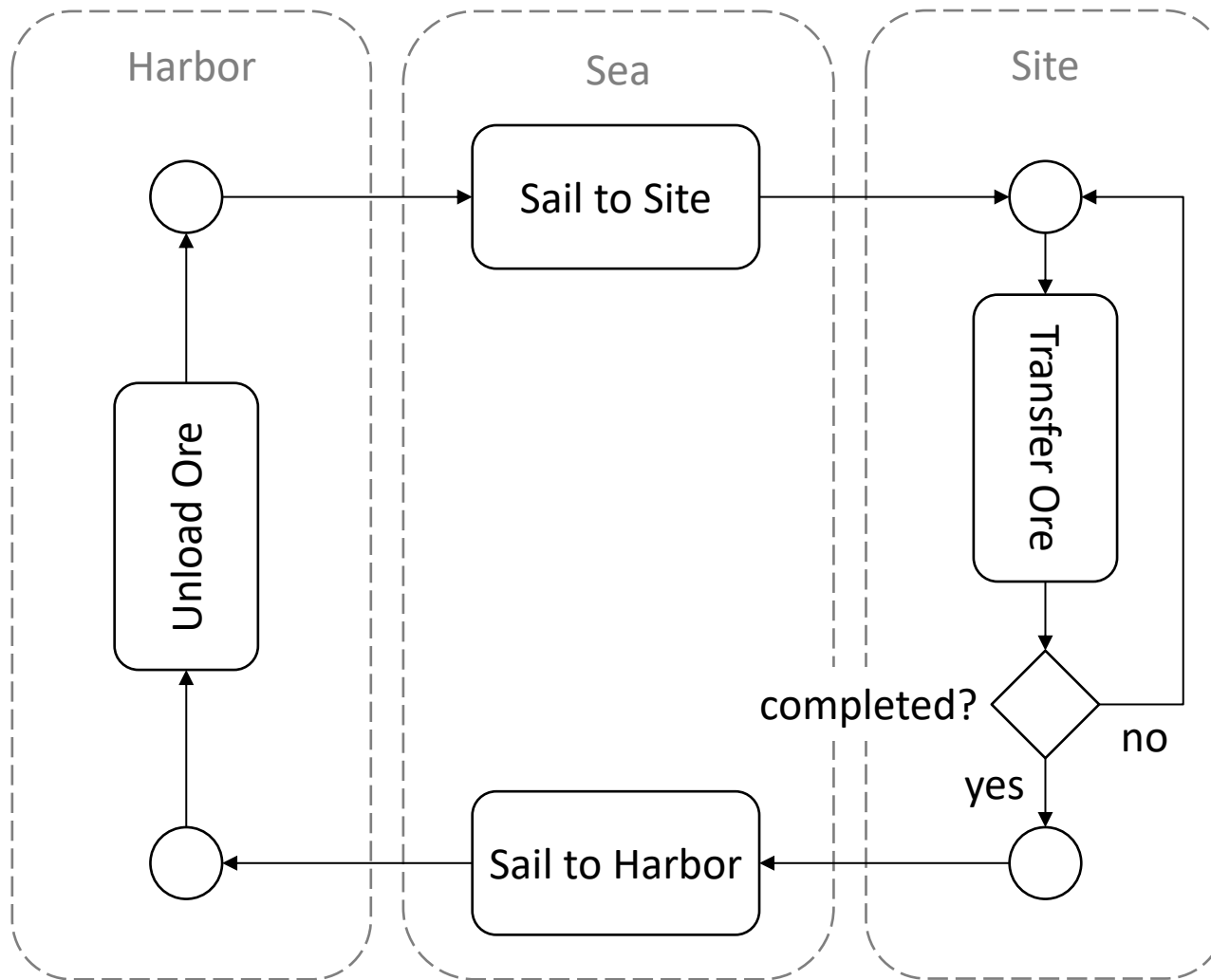
The system changes of mode when some **event** occurs. Events can be either forecasted/planned or unforeseen. The **missions** of the system differ in general from one mode to another one.

The process architecture of the system is conveniently described by **BPMN** or **Harel's State Charts (UML Activity Diagrams)**.

Process Architecture of the Sea Mining System

Actors			Activities
Mining System	Sea	Surface	Update Weather Forecast
		Bed	
	Support Vessel	Mining Tool	Collect Ore
		Lifting System	Lift-Up Ore
		Dewatering System	Dewater Ore
		Transfer System	Transfer Ore
	Bulk Carrier		Sail to Site
			Sail to Harbor
	Harbor		Unload Ore

Bulk Carrier Activities



Activity Tables

	When?	Which resources?	What effects?	Duration
Activity	Trigger	Start	Completion	Duration
Transfer	not transferring dry ore stock > 0 bulk carrier on site cargo < capacity weather forecast OK	transferring = true quantity = min(dry ore stock, max quantity, remaining capacity) dry ore stock -= quantity	transferring = false cargo += quantity	quantity / transfer rate

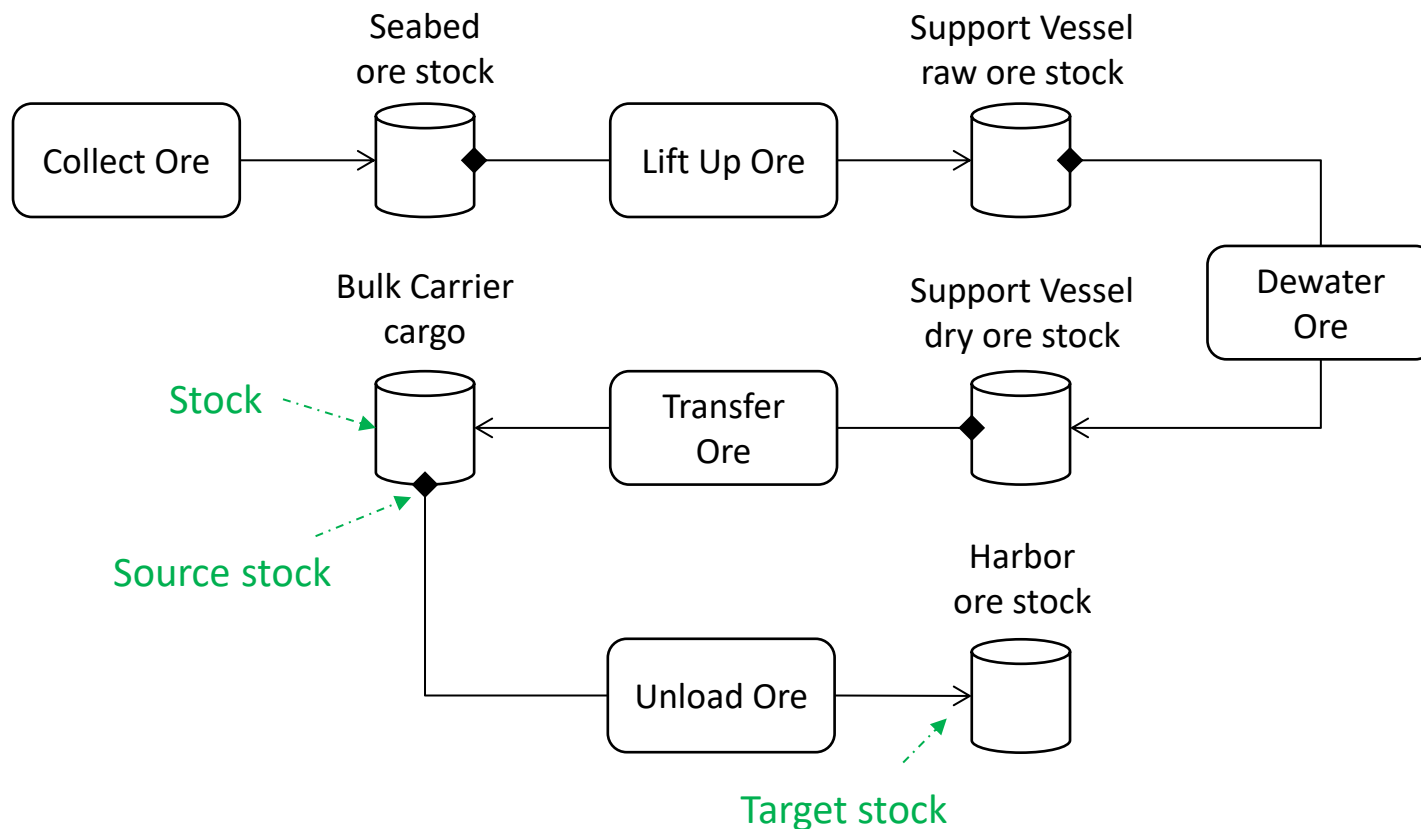
LECTURE 4. THE Σ^{TM} MODELING FRAMEWORK

SECTION 7.3. STOCKS AND FLOWS

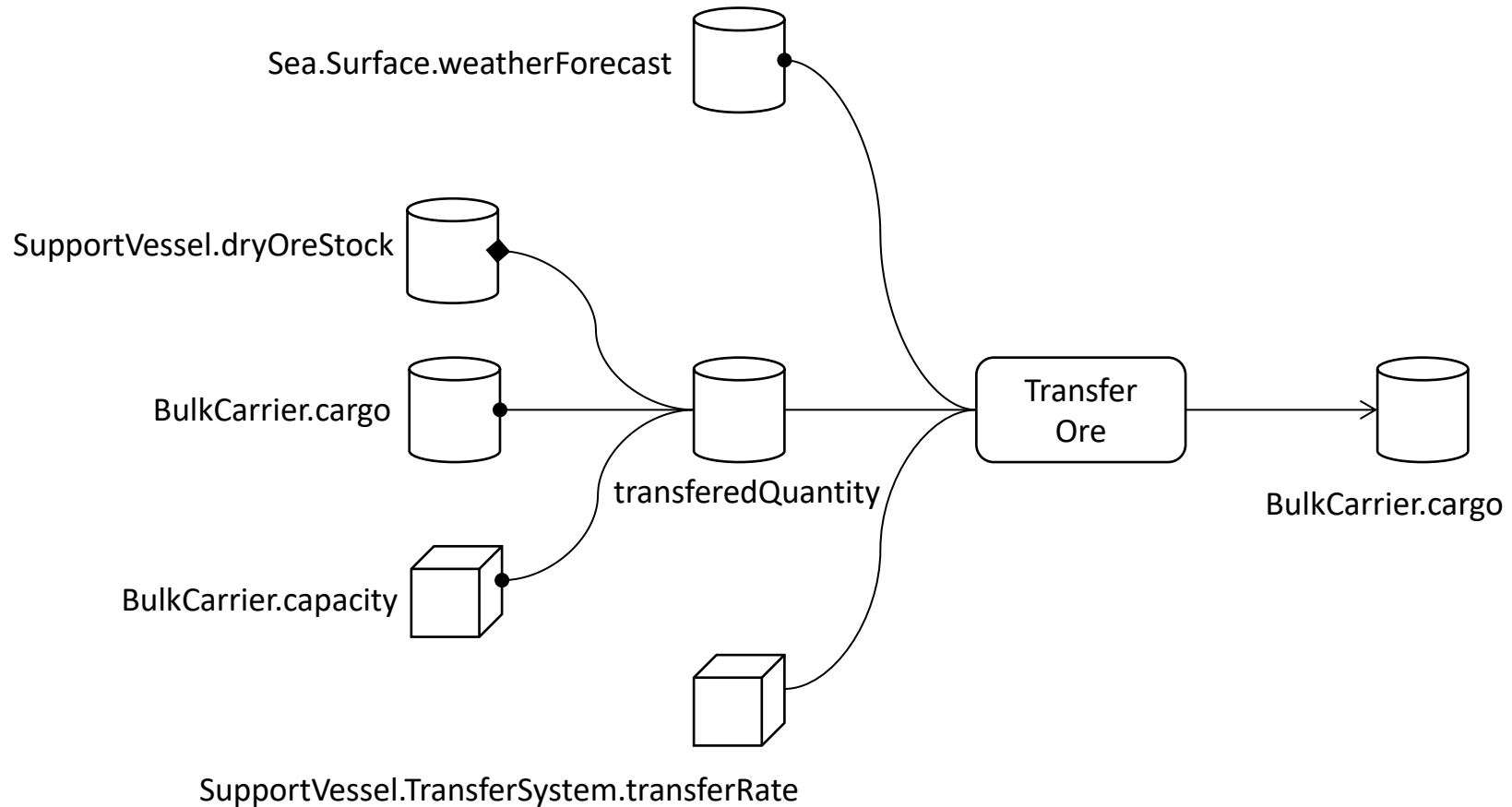
Stocks and Flows

The objective of the "**stocks and flows**" view is to describe the **key stocks** of the system and how these stocks are modified by activities.

Influence Diagrams are useful to do so.



Influence Diagrams



DSM for Mining System

DSM can be used to represent which stock influence which one.

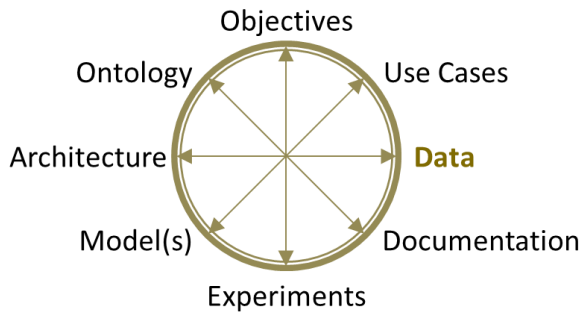
		1	2	3	...	12	...	20	...
1									
2	Sea.Surface.weatherForecast							TO	
3									
...									
12	SupportVessel.dryOreStock							TO	
...									
20	BulkCarrier.cargo							TO	
...									

TO: SupportVessel.TransferSystem.TransferOre

LECTURE 4. THE Σ^{TM} MODELING FRAMEWORK

SECTION 8. DATA

Data



Data are a collection of discrete or continuous values that convey information, describing the quantity, quality, fact, statistics, other basic units of meaning, or simply sequences of symbols that may be further **interpreted formally** [Wikipedia].

The design and operation of any complex technical or socio-technical system requires and produces a lot of data. So does the design and operation of a systemic digital twin of the system.

Most of the times, these data are not gathered in a single place and ready to use. A significant part of the effort required to develop a systemic digital twin consists in **collecting, analyzing** and **formatting data**.

Complex systems are often subject to **uncertainties**. In systemic digital twins, these uncertainties are typically captured using **time series** and **probability distributions**. These are important data to collect, analyze and format.

Operational Data

The design of systemic digital twins is **data intensive**. Collecting relevant data represents a significant part of the development of a systemic digital twin.

Data involved in a systemic digital twin can roughly split into two categories:

- **Constants** and **parameters**: these data are directly embedded in the model.
 - Constants are data that are specified once for all, e.g. the time between two weather forecast (which is decided by meteorological institutes).
 - Parameters are data that can be modified by the analyst or depend on other parameters, e.g. bulk carrier capacity.
- **External data**: these data are embedded in the model via empirical distributions and calls to CSV file or Excel™ spreadsheet.

Collected data, especially external data, often requires some **analysis** and **preprocessing**.

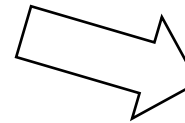
Systemic digital twin designers must have some competences in data analysis.

Weather Forecast

Raw data from the Norwegian Meteorological Institute

```
...  
Rute ukjentFikspostRutetype LinjeLinjefarge Rod4192.699800  
1810.590000 1733126185 BlårammeNavn  
CoastalCodRegulationMTekst 0: Område ID: 2MTekst 1: Fra:  
Bøkfjord fyrMTekst 2: Til: Omgang4204.170000 1810.060020  
1733126185 Blåramme4217.340000 1863.829980 1733126185  
Blåramme4223.190000 1870.249980 1733126185  
Blåramme4223.400000 1869.970020 1733126185  
Blåramme4223.860200 1868.790000 1733126185  
Blåramme4240.549800 1812.720000 1733126185  
Blåramme4242.360000 1805.700000 1733126185  
Blåramme4251.180000 1754.080020 1733126185  
Blåramme4259.299800 1711.500000 1733126185 BlårammeNavn  
CoastalCodRegulationMTekst 0: Område ID: 2MTekst 1: Fra:  
Bøkfjord fyrMTekst 2: Til: Omgang  
...
```

Python
script



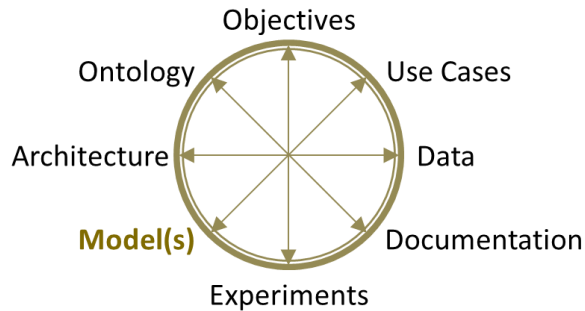
Favorable time window for
ship-to-ship transfer

0	1
6	1
12	1
18	1
24	1
30	1
36	1
42	1
48	1
54	1
60	1
66	1
72	1
78	1
84	1
90	1
96	0
102	0
...	...

LECTURE 4. THE Σ^{TM} MODELING FRAMEWORK

SECTION 9. MODEL(S)

Models



Σ^{TM} models aim at capturing the structure and the dynamics of the systems under study.

However, Σ^{TM} models do not aim at mimicking the physical behavior of the systems.

Models are **abstractions**. An abstraction is relevant if and only if it captures a relevant feature. The objective of designing a model is not to match the reality, but to **learn** something about the system under study.

The design of a model is **highly iterative**. Each iteration consists in adding a feature to the model, then to test it extensively. The smaller the iterations, the most efficient the design.

The development of a systemic digital twin requires in general the design of several Σ^{TM} models. It is in particular a good idea to **test features separately**, within a dedicated small models.

Dissolve the target problem in the rising tide of general results.

User Interfaces

The experience shows that three categories of **stakeholders** are involved in the design and the use of systemic digital twins:

- The **designers** who have competences in modeling and simulation (often with a consulting company).
- The **specialists** who have competences in the application domain of the systemic digital twin (often with the client company).
- The **executive sponsors**, from the client company, who finance the design.

The designers and the specialists are interested in the **design** of the systemic digital twin. The executive sponsors on the contrary are mainly interested in the **results** obtained from the systemic digital twin.

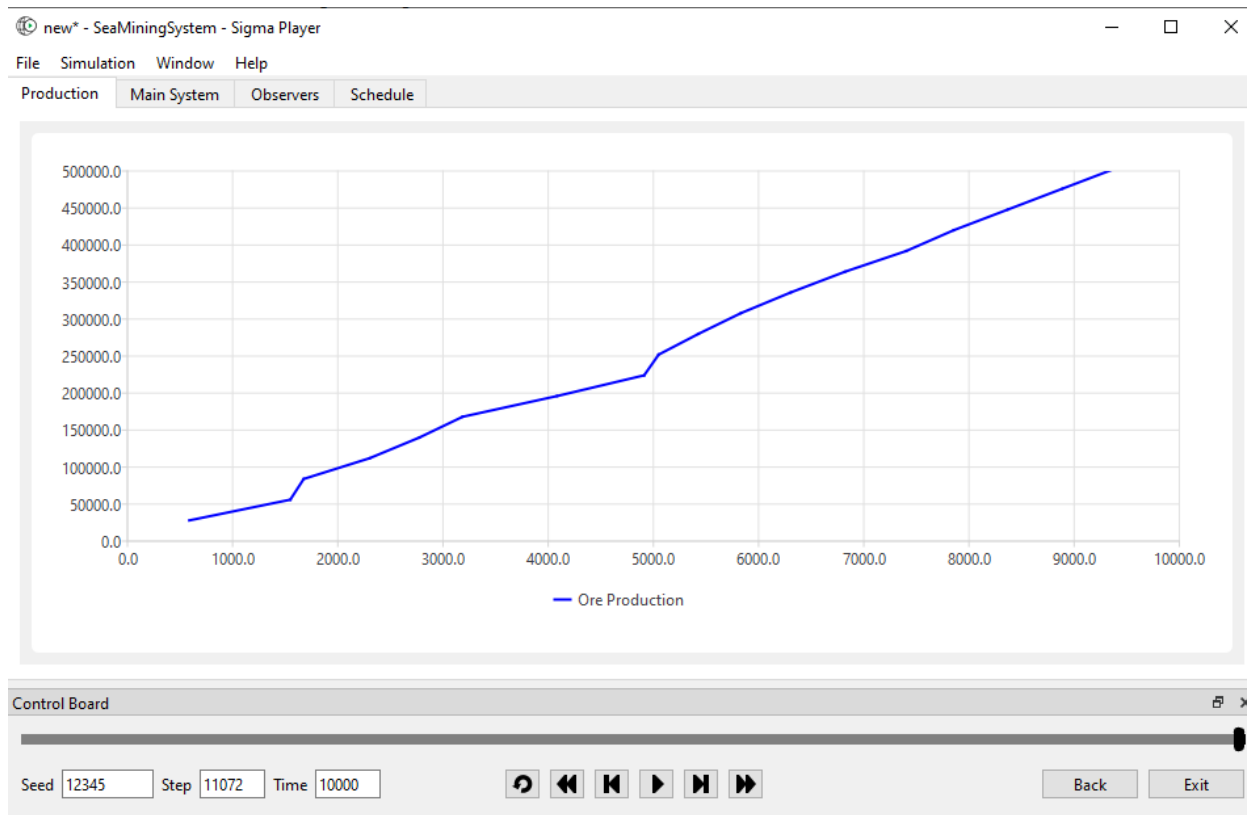
The user interface of the systemic digital twin should accommodate these two types of users:

- In depth analysis of the simulation to track **modeling and calibration errors**.
- Presentation of **key performance indicators** in a manager friendly manner.

The latter is fully part of the design.

Mining System

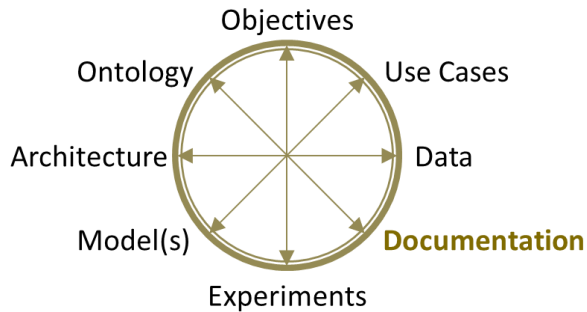
The WIDL language (Lecture 6) makes it possible to design ad hoc graphical simulation interfaces.



LECTURE 4. THE Σ^{TM} MODELING FRAMEWORK

SECTION 10. DOCUMENTATION

Documentation



A systemic digital twin must be self-contained. It should integrate all the **artifacts** (documents, data, programs, models...) required for its design, as well as all artifacts it produces throughout its lifecycle.

Collecting and organizing the documentation of a system digital twin is essential. Recall that the main objective of a systemic digital twin is to learn about the system under study and to **transmit** and **capitalize** the acquired **knowledge**. Without a solid documentation this is just impossible.

The adage “*a picture is worth a thousand words*” rightly highlights the expressive power of images, diagrams, and other graphical representations. Nevertheless, when the objective is to formulate a precise, unambiguous, and formally verifiable description, written language (text) remains the indispensable medium.

Last but not least: a good documentation is a **concise** documentation. The elegance is reached when everything unessential has been removed.

Recommendations

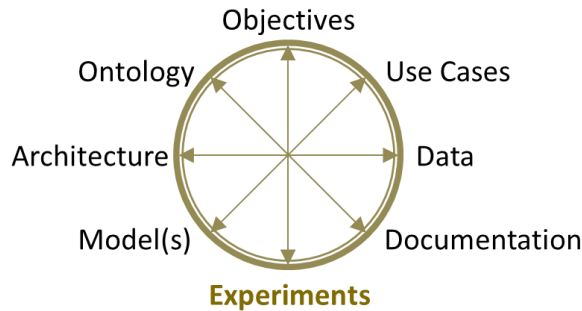
All the artifacts designed in the previous sections are legitimate to be part of the documentation of the systemic digital twin.

- Page cover with a presentation of the system under study and the objectives of the systemic digital twin.
- Key metaphor
- Ontology
- Uses Cases
- Logical and Process Architecture
- Stocks and Flows
- Data
- Examples of experiments

LECTURE 4. THE Σ^{TM} MODELING FRAMEWORK

SECTION 11. EXPERIMENTS

Experiments



In system dynamics, **experiments** consists mainly in performing **interactive** or **stochastic simulations**. These are experiments **in silico**.

Experiments in silico are nevertheless experiments. As such they must obey **rigorous experimental protocols**:

- Hypotheses must be clearly set up.
- Operations to perform must be clearly defined.
- Conclusions drawn for the system must follow from the results of experiments on the model.

Moreover, they must be (easily) **reproducible**.

Concretely, simulations of Σ^{TM} models involve two types of **parameters**: parameters of the model itself and parameters of the simulation. Both must be carefully managed for the experiment to be of interest.

LECTURE 4. THE Σ^{TM} MODELING FRAMEWORK

SECTION 12. WRAP-UP

Main Difficulties of Designing a Systemic Digital Twin

Difficulty 1: Dive into the **solution** before stating the **problem**.

Take the time to state the problem is the only way to ensure that nothing has been forgotten and that all potential solutions have been considered.

Difficulty 2: **Overload** the analysis with useless considerations and/no being unable to define correctly the **hierarchy of concerns**.

The objective of a systemic digital twin is not to mimic accurately the system under study but to **make decisions** about its design and operation.

Do not hesitate to design simplified models as a means of communication and validation of your approach.

Difficulty 3: **Reason too abstractly** and/or **make overcommitting assumptions**.

There is a permanent risk of falling into “**abstract non-sense**” and to make assumptions that do not correspond to the “reality”. It is of primary importance to always go back to **concrete issues**, typically by designing **use cases**.

Difficulty 4: Being stuck due to the **lack of data**.

Wrap-Up

- A model should be seen a part of a **constructive** and **pragmatic proof** that it is possible to design, to build and to operate a system having some specified properties. These properties are formalized as **requirements**.
- The design a **systemic digital twin** is an **iterative process**.
- The Σ^{TM} **modeling framework** aims at designing systemic digital twins efficiently and accurately using the Σ^{TM} technology. It consists of **8 activities**:

