OPM Primer

[TE-265][LEC-003]

Time

Formulation

Pre-Phase A (Phase 0 to ECSS) – Concept Studies

Phase A – Concept and Technology Development

Phase B – Preliminary Design and Technology Completion

Implementation

Phase C – Final Design and Fabrication

Phase D – System Assembly, Integration and Test, Launch

Phase E – Operations and Sustainment

Phase F – Closeout

Problem

Napkin Model

Operational Analysis

System Analysis

Logical Architecture

Physical Architecture

EPBS

Solution

Using MBSE through the other Phases

MDE

System Engineering

MBSE

OPM

ARCADIA

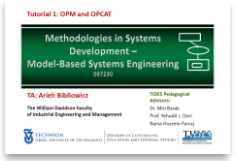
Related Analysis

Future Perspective

DATE	SES-#	CLASS	DELIVERABLES
	001	[LEC-000] Course Introduction – 1h [PRD-001] CASE RESEARCH Definition [LEC-001] MBSE Introduction: SE, MDE [AGN-001] MBSE Methodologies	
	002	Meeting: [PRD-001] & [AGN-001]	[AGN-001] [PRD-001]
	003	[LEC-002] STK/MIS Requirements [LEC-003] OPM Primer [AGN-002] OPM: Things & Structure [AGN-003] OPM: Processes & Behavior	
	004	Meeting: [AGN-002] & [AGN-003]	[AGN-002] [AGN-003]
	005	[LEC-004] CONOPS [PRD-002] CASE CONOPS Definition [AGN-004] OPM: Complexity Management & Logical Operations & Simulations	
	006	Meeting: [AGN-004] [PRD-002]	[PRD-006] INTRO / LIT [PRD-002] [AGN-004]
	007	[LEC-005] System Requirements [PRD-004] Requirements ** Review of info to CONCEPT REPORT	
	008	**[TST-001] CONCEPT MODELLING & OPM QUESTIONS** [PRD-003] CONCEPT Presentation	[PRD-003] [PRD-006] CONCEPTUAL SECTION

References for this section

- Main:
  - REF-005: DORI, D. **Model-Based Systems Engineering with OPM and SysML**. New York: Springer, 2016. ISBN 978-1-4939-3294-8.
- OPM Starting Guide
  - http://esml.iem.technion.ac.il/introduction-to-opm/



The following classes have “parts” of the presentations / papers of the OPM Starting Guide references and the ISO

References for this section

- Main:
  - REF-005: DORI, D. Model-Based Systems Engineering with OPM and SysML. New York: Springer, 2016. ISBN 978-1-4939-3294-8.
- OPM Starting Guide
  - http://esml.iem.technion.ac.il/introduction-to-opm/
- https://opcloud-trial.firebaseapp.com/

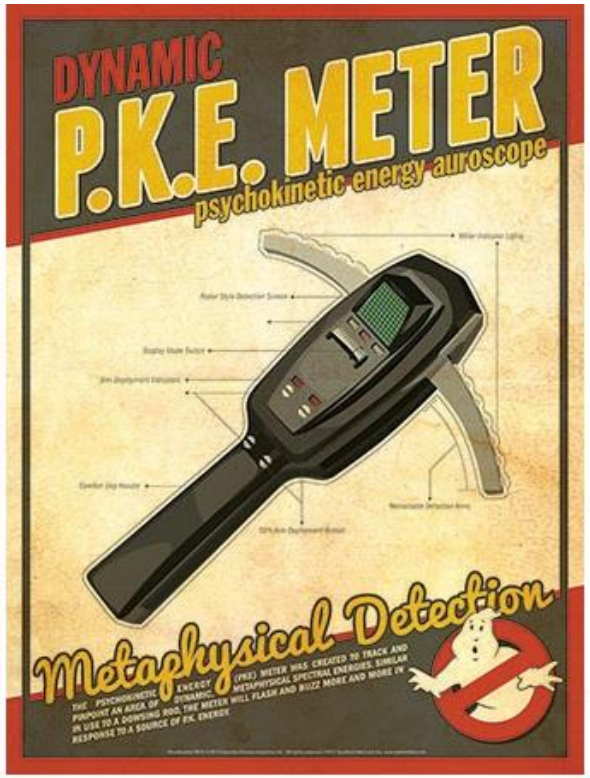
The following classes have “pieces” of the presentations/papers of the OPM Starting Guide

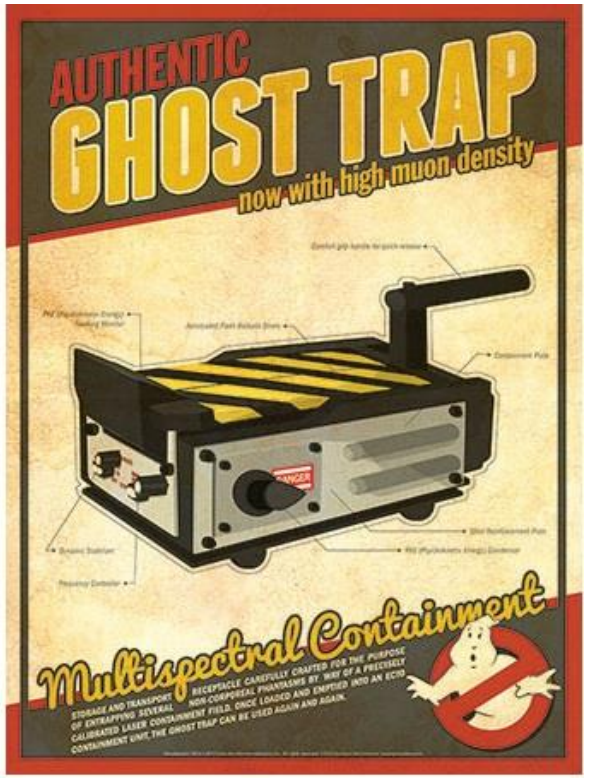
DRIVING PROJECT

Ghost Catching

Ghostbusters

Ghost Buster Tools

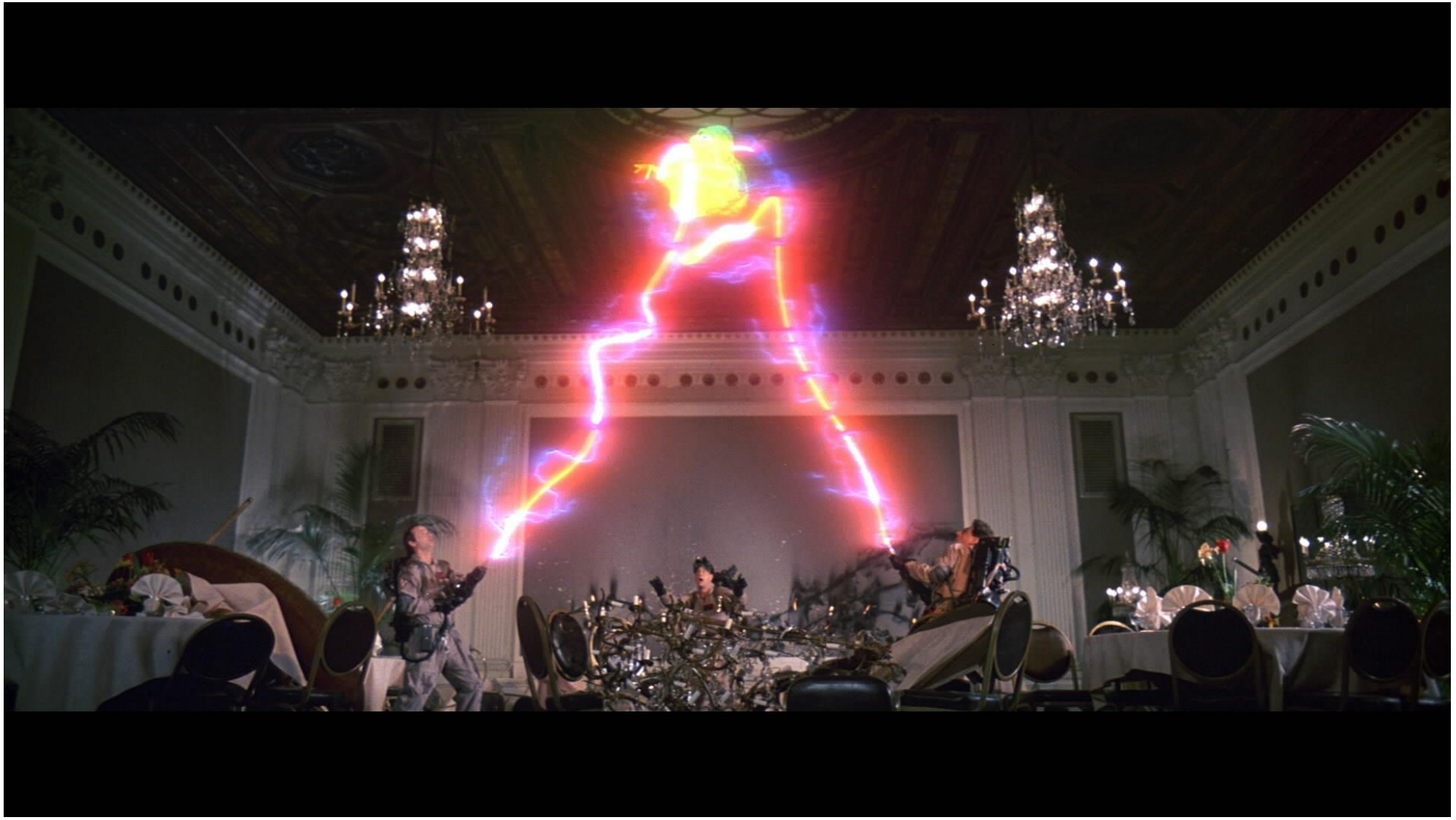


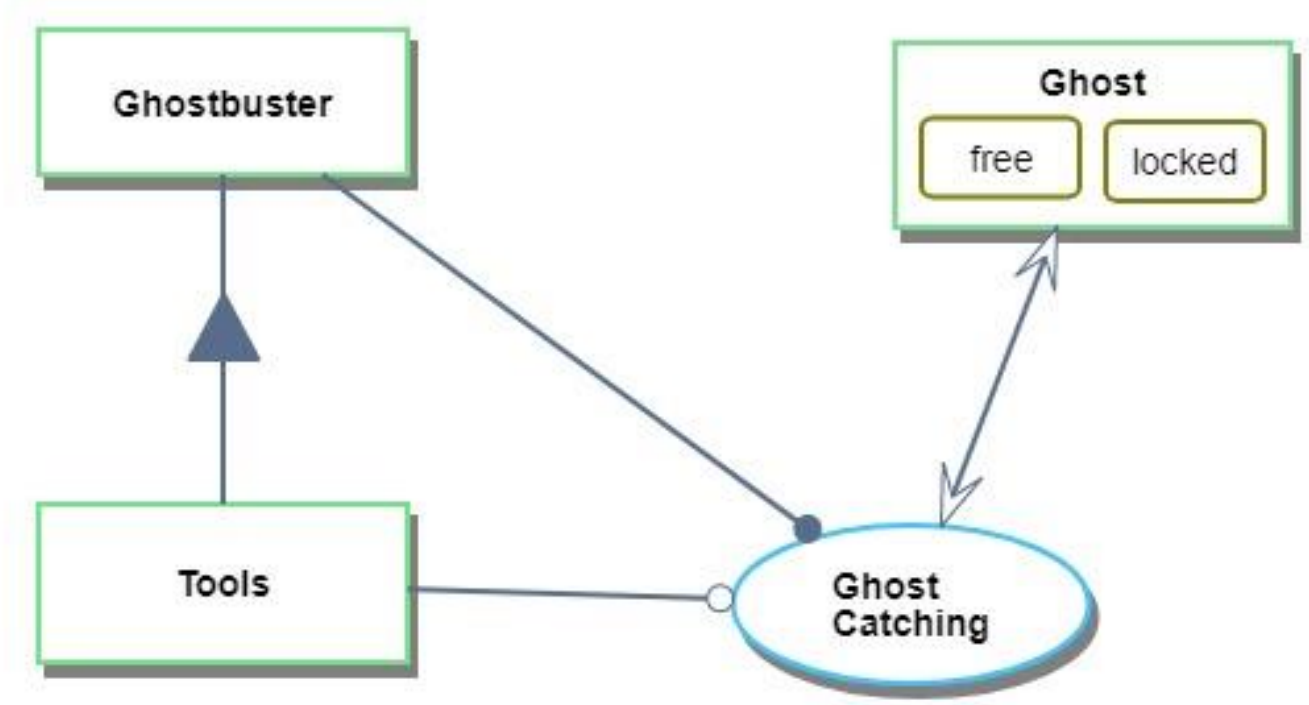


ECTO-01

Slimer Ghost

Catching





Ghostbuster is physical and systemic.
Ghost is physical and systemic.
Ghost can be free or **locked**.
Tools is physical and systemic.
Ghostbuster consists of Tools.
Ghost Catching is physical and systemic.
Ghostbuster handles Ghost Catching.
Ghost Catching requires Tools.
Ghost Catching affects Ghost.

OPM FOUNDATIONS

Conceptual Modelling

Convert — fragmented knowledge into explicit and integrated knowledge

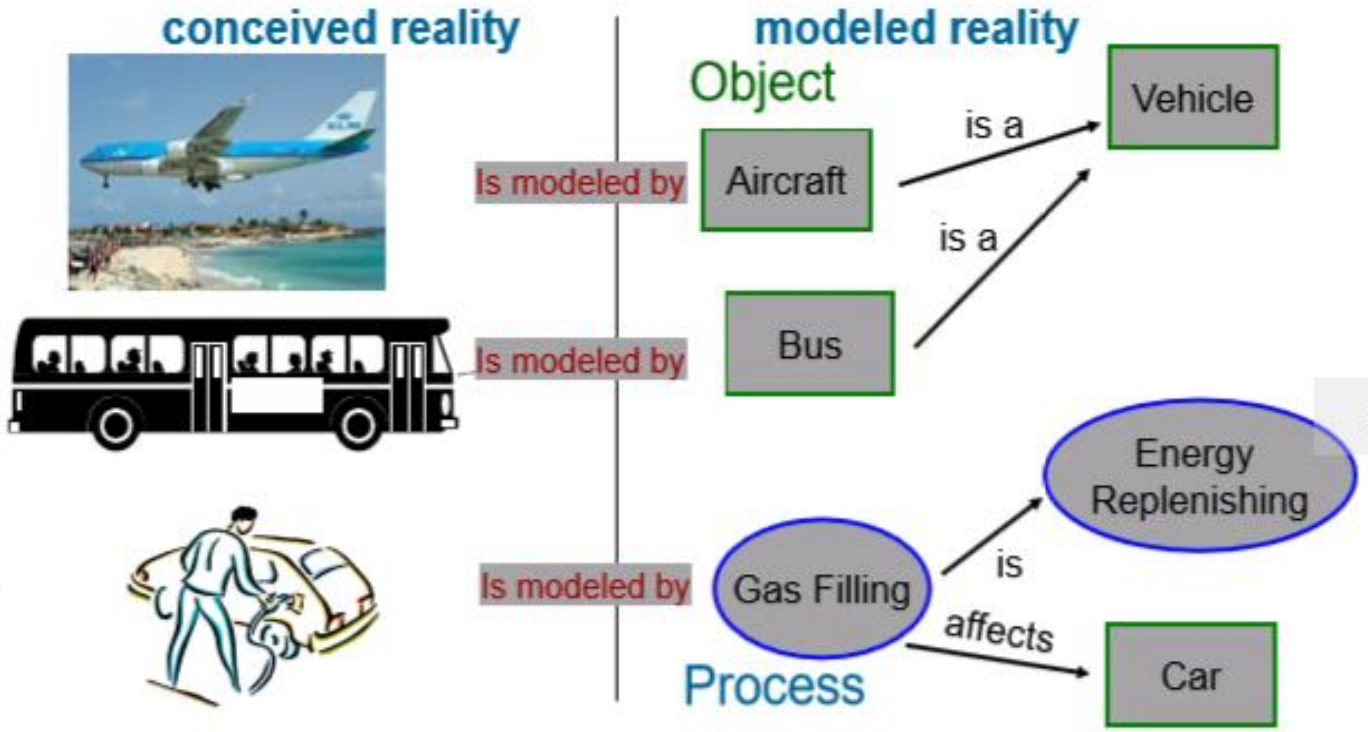
+

Construct — concise models

+

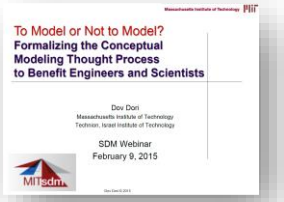
Communicate — The model to stakeholders

→ **Concept Models**



- **simple** yet **expressive**, and
- **intuitive** yet **formal**

16:23



[Dori] Fundamental Questions:

- **1. What is needed to describe the universe?**
  - "Things" and their "relations"
- **2. What can those things do?**
  - Things can exist or happen.
- **3. What are the things that exist in the world?**
  - Objects exist – statics (time-independent).
- **4. What are the things that happen in the world?**
  - Processes happen – are dynamics (time-dependent).

[Dori] Fundamental Questions:

- **5. How do objects and processes relate?**
  - Processes happen to objects. While happening,
  - Processes transform objets.
- **6. Transform?? what does that mean?**
  - Create
  - Destroy
  - Affect
    - an object

[Dori] Fundamental Questions:

- **7. Affecting? What does that mean?**
  - A process affects an object by changing its state. Hence, objects must have states.
- **8. What are the two major aspects of any system?**
  - Structure: static aspect – what the system is made of?
  - Behaviour: dynamic aspect - how the system changes over time?
- **9. Which third aspect is specific to man-made systems?**
  - Function: the utilitarian, subjective aspect. Why? for whom? Who benefits?

[Dori] Object-Process Theorem

Objects with **states**, **processes** and their relations among them constitute a **necessary** and **sufficient universal ontology** to describe a **system**.

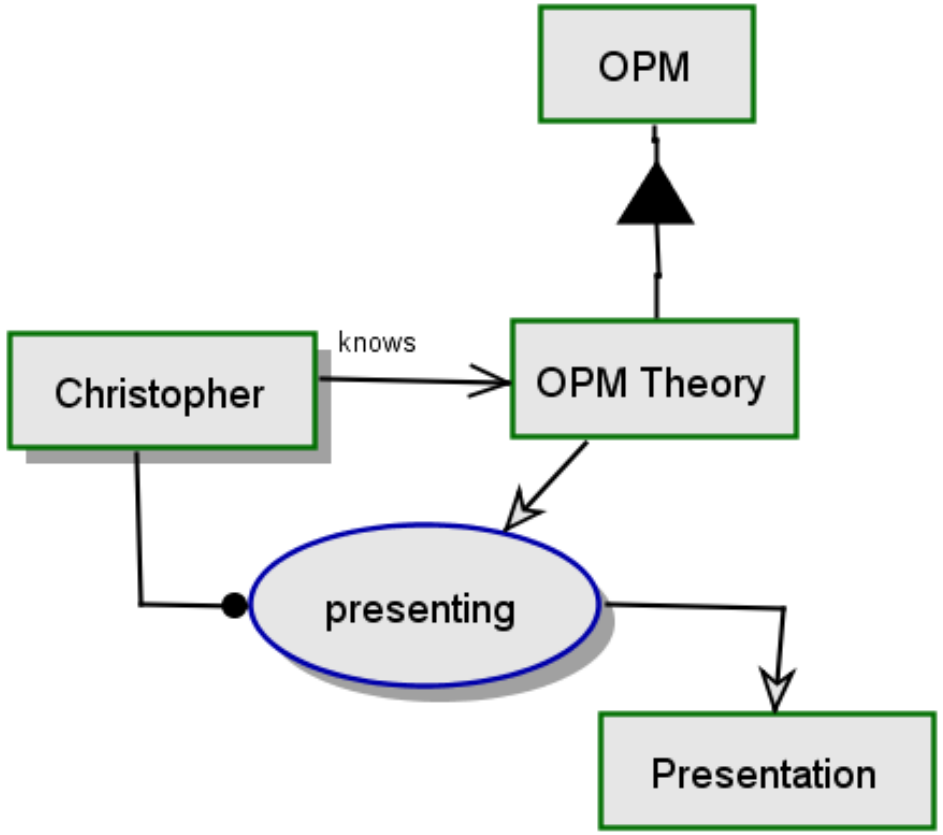
Majors aspects of any system?

- **Structure** – the static aspect. **What** the system is made of.
  - *Time-independent*
- **Behavior** – the dynamic aspect. **How** the system changes over time.
  - *Time-dependent*

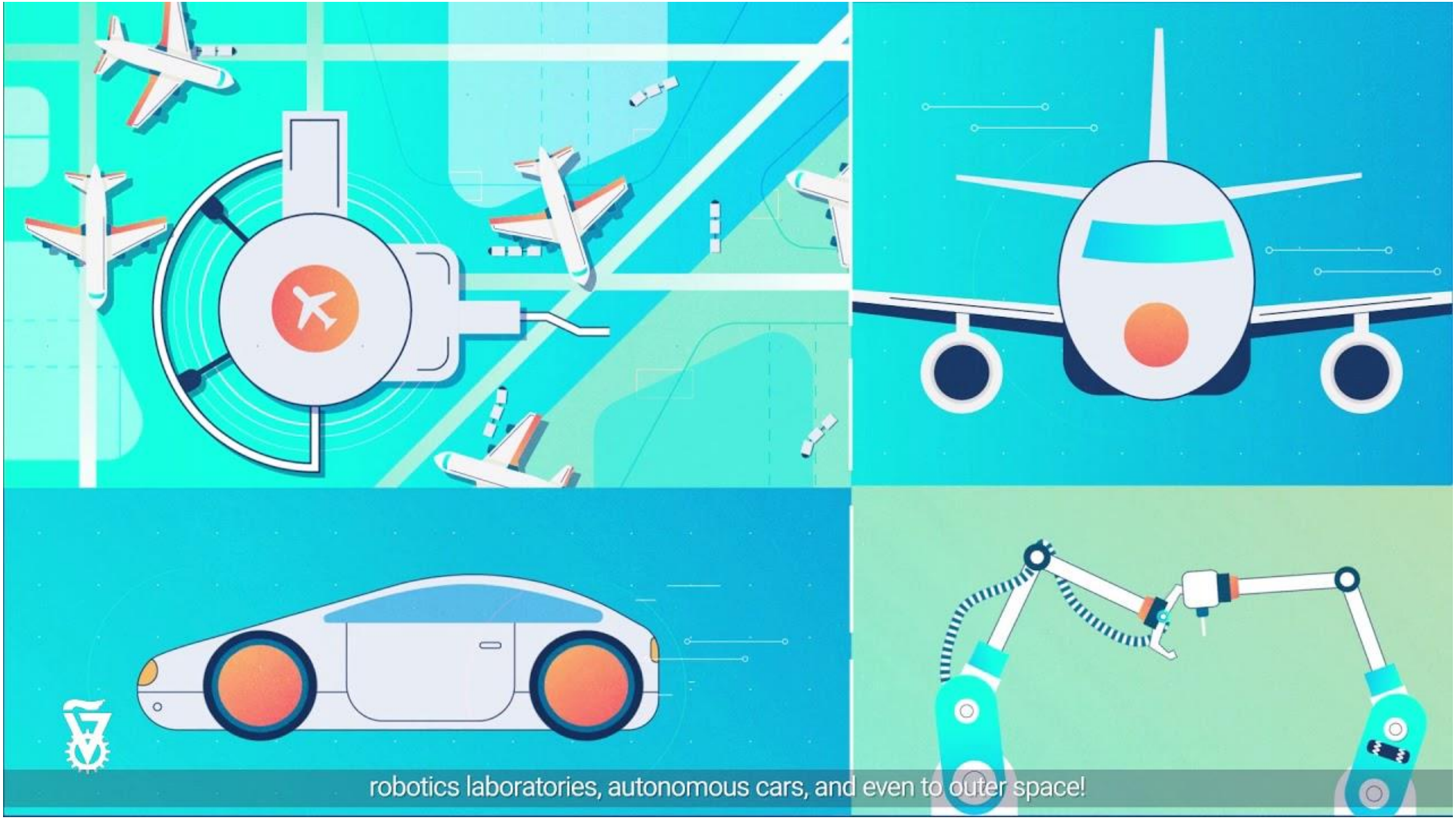
Only one Diagram Type

- **Function** – The utilitarian, subjective aspect.
  - **Why** is the system built?
  - For **whom** is the system built?
  - **Who** benefits from operating the system?

Cognitive channels: visual-OPD and textual-OPL



Christopher is physical.
Christopher knows OPM Theory.
Christopher handles presenting.
OPM consists of OPM Theory.
presenting is physical.
presenting consumes OPM Theory.
presenting yields Presentation.


robotics laboratories, autonomous cars, and even to outer space!

12 OPM Principles

- **1. The Function-as-a-Seed** – Modelling a system starts by defining, naming, and depicting the function of the system, which is also its top-level process.
- **2. The Model Fact Representation** – An OPM model fact needs to appear in at least one OPD in order for it to be represented in the model.
- **3. The Timeline** – The timeline wihin an in-zoomed process is directed by default from the top of the in-zoomed process elipse to its bottom

- **4. The Minimal Conceptual Modelling Language** – A symbol system – a language – that can conceptually model a given system using ontology with fewer diagram kinds and fewer symbosl and relations among them is prefeable over a larger ontology with mode diagram kinds and more symbols and relations among them.
- **5. The Thing Importance** – The importance of a thing T in an OPM Model is directly related to the highest OPD in the OPD hierarchy where T appears.

- **6. The Object Transformation by Process** – In a complete OPM Model, each process must be connected to at least one object that the process transforms or one state of the object that the process transforms.
- **7. The Procedural Link Uniqueness** – At any level of detail, na object and a process can be connected with at most one procedural link, which uniquely determines the role of the object with respect to the process.
- **8. The Singular Name** – A name of an OPM thing must be singular. Plural has to be converted to singular by adding the world “Set” for inanimate things or “Group” for humans.

- **9. The Graphics-Text Equivalence** – Any model fact expressed graphically in an OPD is also expressed textually in the corresponding OPL paragraph.
- **10. The Thing Name Uniqueness** – Different things in na OPM Model which are not features must have different names. Features are distinguishable by appending to them the reserved word “of” and the name of their exhibitor.
- **11. The Detail Hierarchy** – Whenever na OPD becames hard to comprehend due to an excessive amount of details, a new, descendant OPD shall be created.
- **12. The Skip Semantics Precedence** – Skip semantics takes precedence over wait semantics.

BUILDING BLOCKS

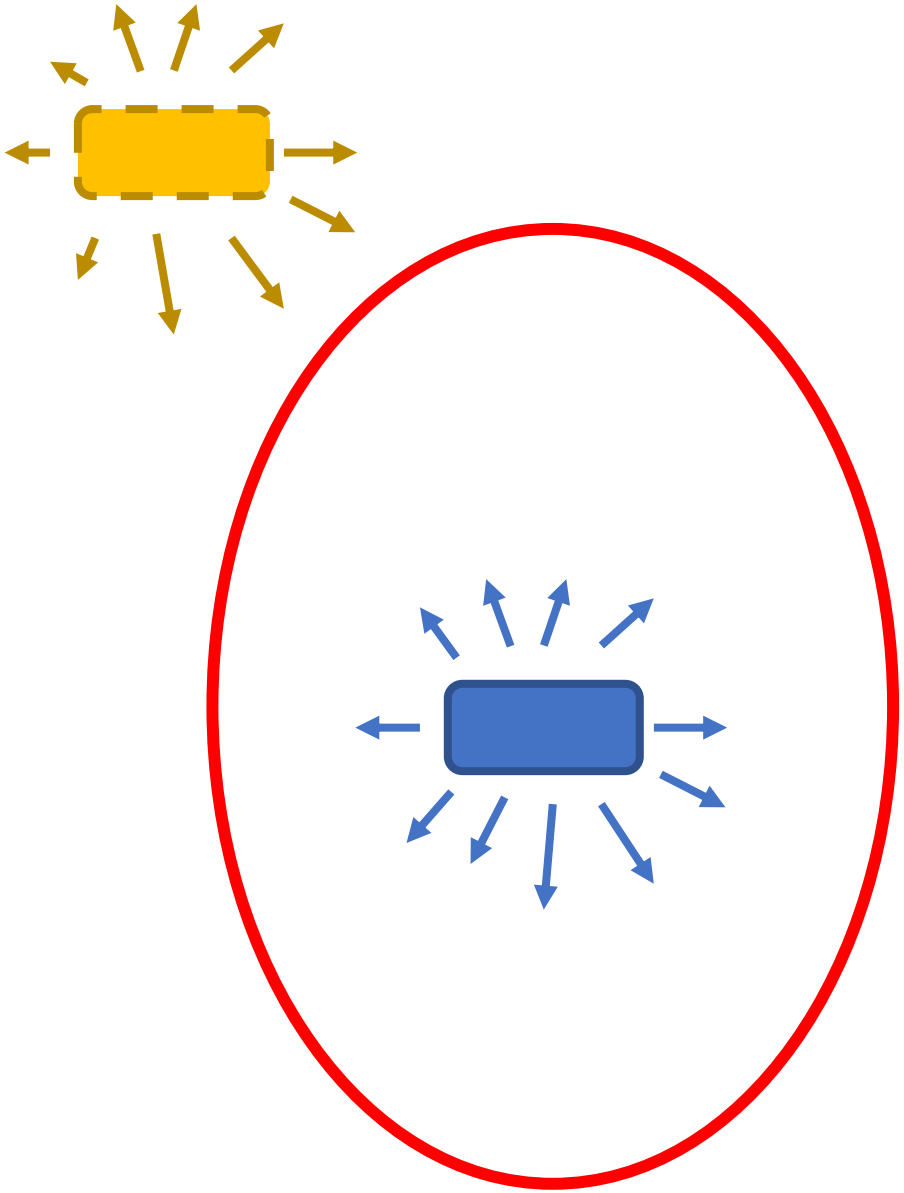
Essence

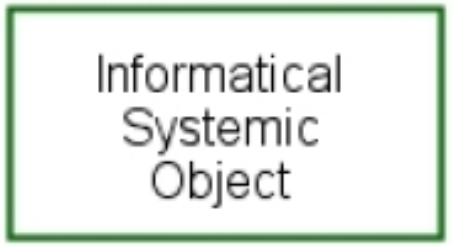
- **Key to modeling Cyber-Physical Systems**
  - **Physical** objects in the model represent what is really "out there" –actual states and values of objects
  - **Informatical** objects represent information about their corresponding physical objects

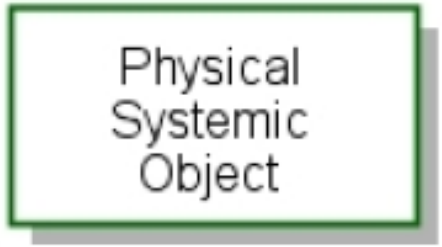
- Only ***informatical objects*** are available to a decision-making agent (human or artificial)

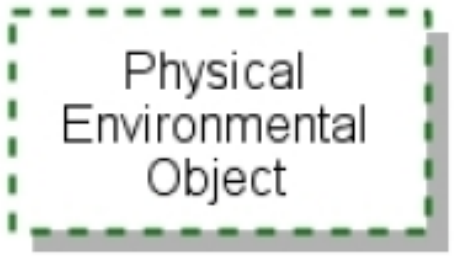
Affiliation

- **Affiliation**, which pertains to the thing's scope and denotes whether the thing is:
  - **systemic**, i.e. part of the system, or
  - **environmental**, i.e. part of the system's environment.







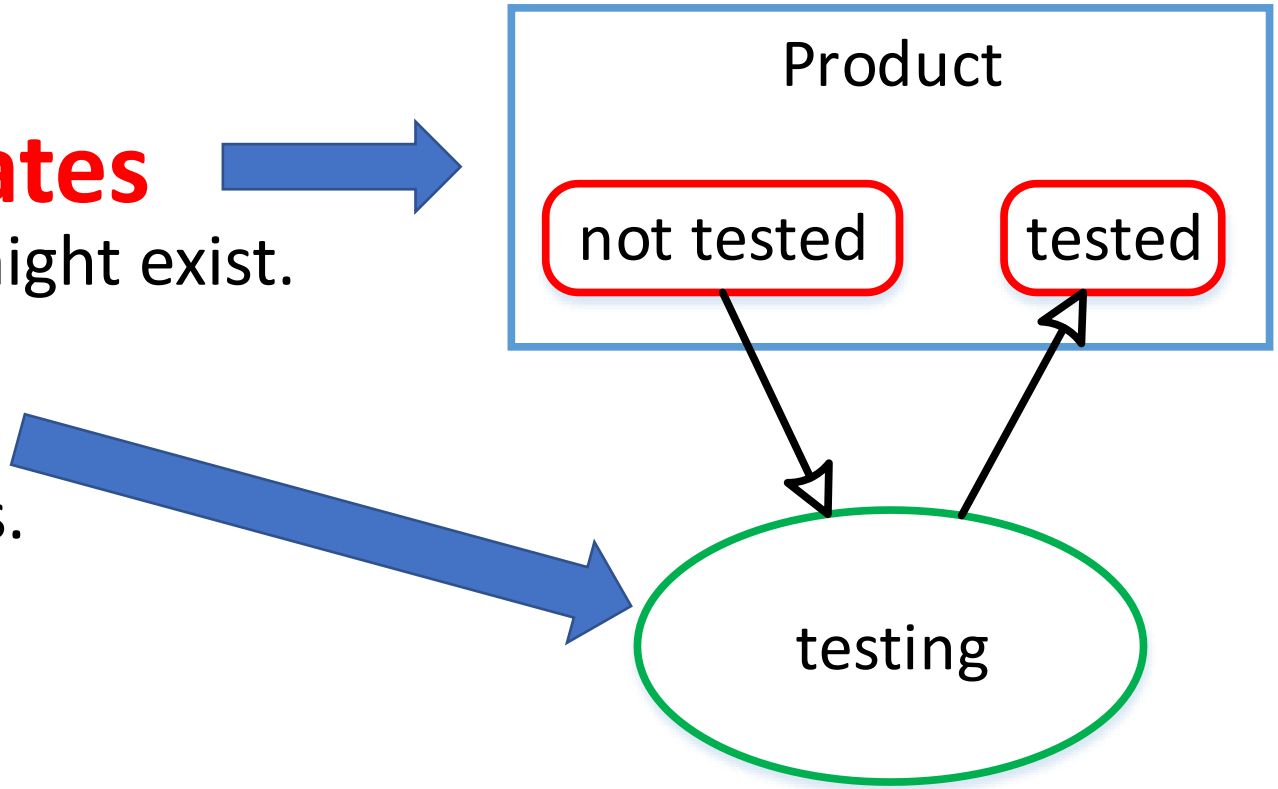


Informatical Systemic Process is an informatical and systemic process.
Physical Systemic Process is a physical and systemic process.
Informatical Systemic Object is an informatical and systemic object.
Physical Systemic Object is a physical and systemic object.
Informatical Environmental Process is an informatical and environmental process.
Physical Environmental Process is a physical and environmental process.
Informatical Environmental Object is an informatical and environmental object.
Physical Environmental Object is a physical and environmental object.

OPM has only two building blocks:

1. **Objects with states**
   A thing that exists or might exist.

2. **Processes**
   A thing that transforms.



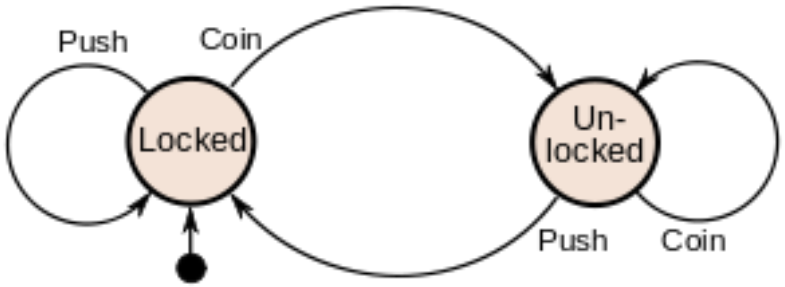
All the other elements are **relations** between things, expressed graphically as links

A Object Thing

- An **object** is a **thing** that exists. Its existence can be either physical or conceptual. An **object** is a **thing** that can be transformed.
- An **object** can represent simple things such as car keys, or complex systems such as manufacturing plants.
- The graphical representation of an **object** in
- OPM is a square:

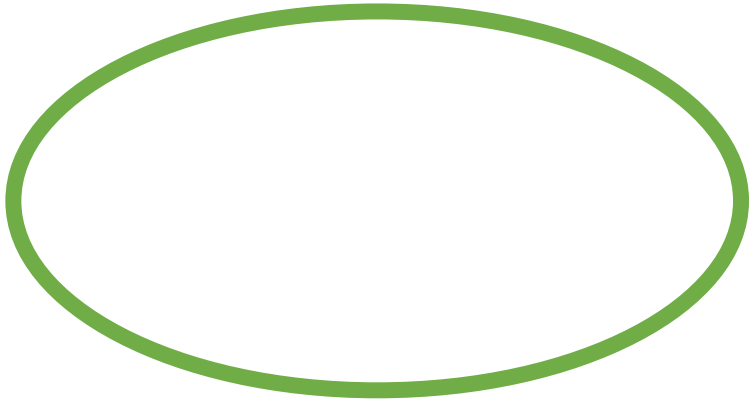
States and Values

- A **state** is a possible situation at which an object can be.
- A **state** is only meaningful in within the context of a containing object.
- The graphical representation of a **state** in OPM is a rountangle (rounded rectangle):



A Process Thing

- A **process** is a thing that transforms an **object**.
- This transformation can be:
  - Creation of an **object**.
  - Consumption of an **object**.
  - Changing the state of an **object**.
- By definition, a process must be associated with at least one object.
- The graphical representation of a process in OPM is an ellipse:

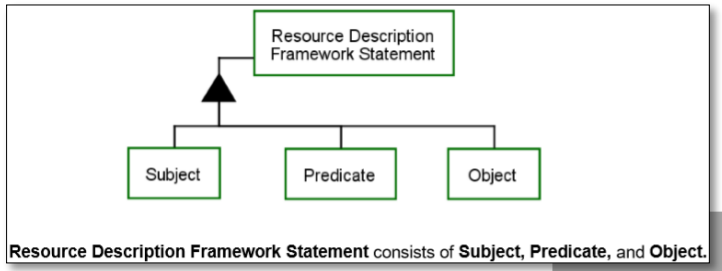


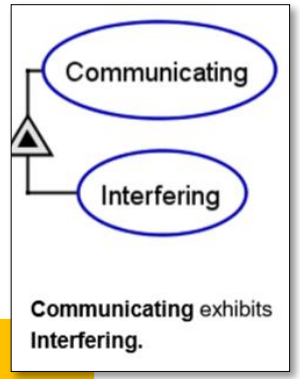
STRUCTURE

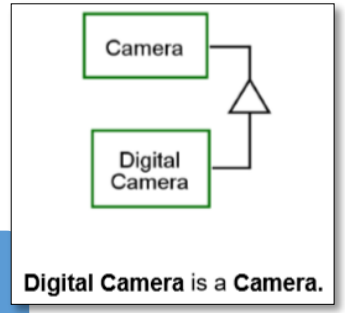
Structural Links

- **Structural Link** is a link that specificies **a static aspect** of the system by connecting na object to another object or a process to another process.

Structural Links







Aggregation

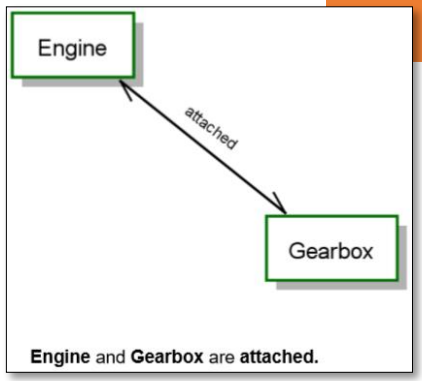
Exhibition

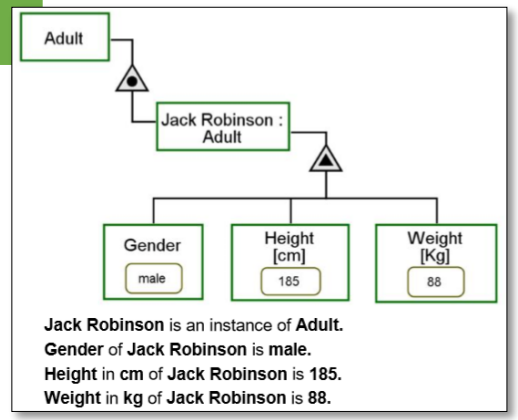
Specialization

Relational

Structural

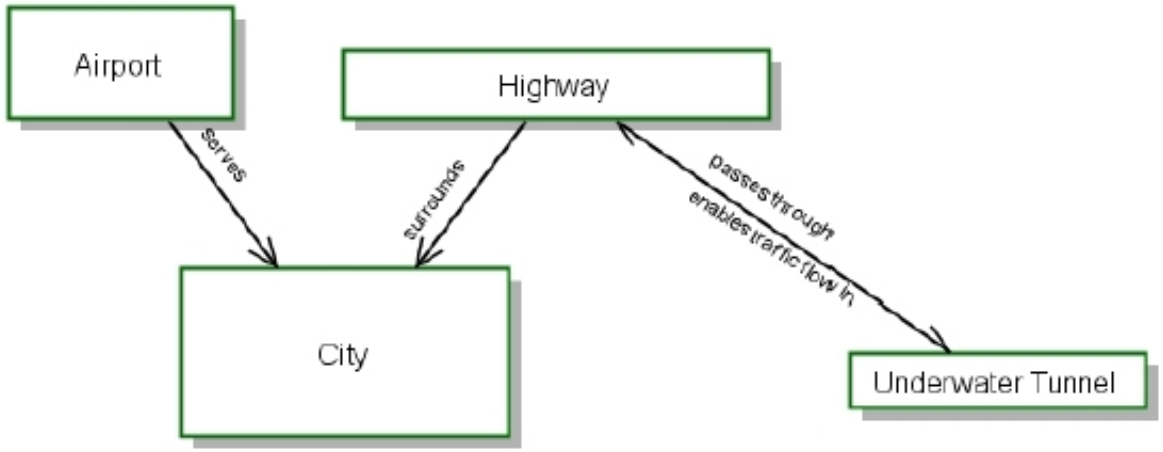
Instantiation



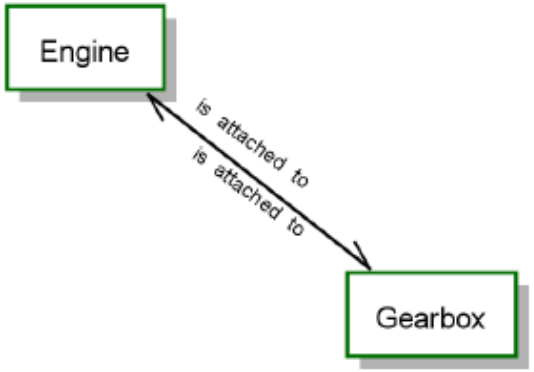


Tagged Structural Links

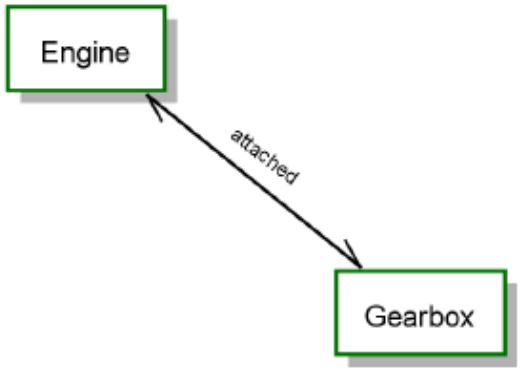
- A unidirectional tagged structural link defines a structural link between a source **object** and a **target** object.
  - The syntax of the unidirectional tagged structural link OPL sentence shall be: ***Source-thing tag Destination-thing***.
  - The syntax of the unidirectional null-tagged structural link OPL sentence shall be: ***Source-thing*** *relates to* ***Destination-thing***.
  - The syntax of the reciprocal tagged structural link with only one tag shall be: ***Source-thing*** *and* ***Destination-thing*** *are* ***reciprocity-tag***.
  - The syntax of the reciprocal tagged structural link with no tag shall be: ***Source-thing*** *and* ***Destination-thing*** *are related.*



Airport serves City.
Highway surrounds City.
Highway passes through Underwater Tunnel.
Underwater Tunnel enables traffic flow in Highway.



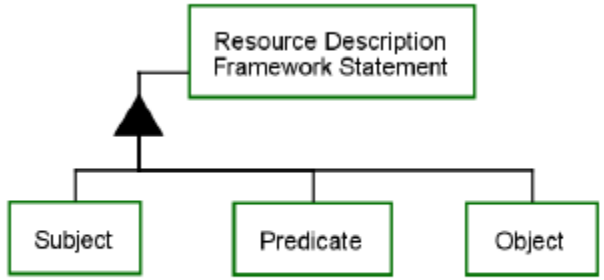
Engine is attached to Gearbox.
Gearbox is attached to Engine.



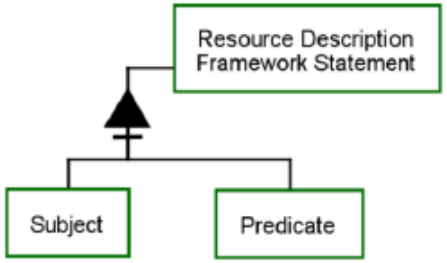
Engine and Gearbox are attached.

Aggregation-Participation

- A structural relation which denotes that one (high level) **thing** aggregates (i.e. consists of, contains) one or more (low level) **things**. The high-level **thing** is called the *whole* or the *aggregate* while the lower-level **things** are called the *parts*.
  - The syntax of the aggregation-participation relation link shall be: ***Whole-thing*** *consists of* ***Part-thing1***, ***Part-thing2***, …, *and* ***Part-thingn***.



Resource Description Framework Statement consists of **Subject**, **Predicate**, and **Object**.



Resource Description Framework Statement consists of **Subject**, **Predicate**, and at least one other part.

Exhibition-Characterization

- A structural relation which denotes that one **thing** exhibits (or is characterized) by another **thing**. A **thing** exhibits *features* that characterize it: An *attribute* is a *static feature* & An *operation* is a *dynamic feature*.
- The main difference between exhibition and aggregation is that an attribute always has a value, whether a part may be inexistent: A bag of candies will be empty after Purim (aggregation), but it will always have a color (exhibition).
  - The syntax of the exhibition-characterization relation link for an object exhibitor with a complete collection of n attributes and m operations shall be: ***Object-exhibitor exhibits Attribute1, Attribute2, … , and Attributen,*** *as well as* ***Operation1, Operator2, … , Operatorm.***
  - The syntax of the exhibition-characterization relation link for a process exhibitor with a complete collection of n operation features and m attribute features shall be: ***Process-exhibitor exhibits Operation1, Operator2, … , Operatorn,*** *as well as* ***Attribute1, Attribute2, …, and Attributem.***

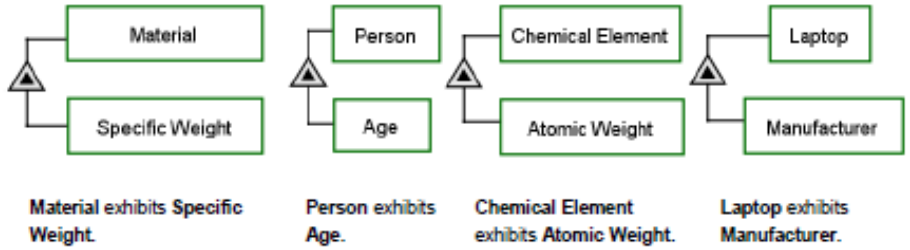


Figure 20 — Object attribute examples

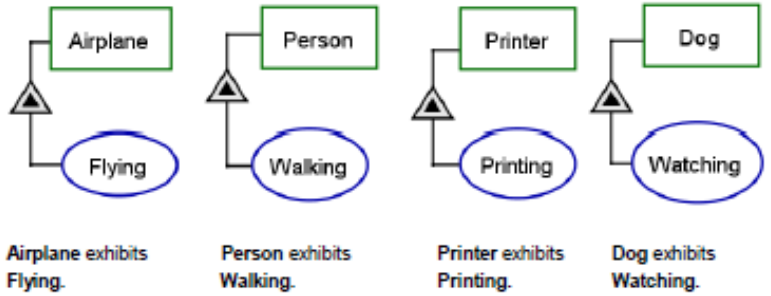


Figure 21 — Object exhibitor with operation examples

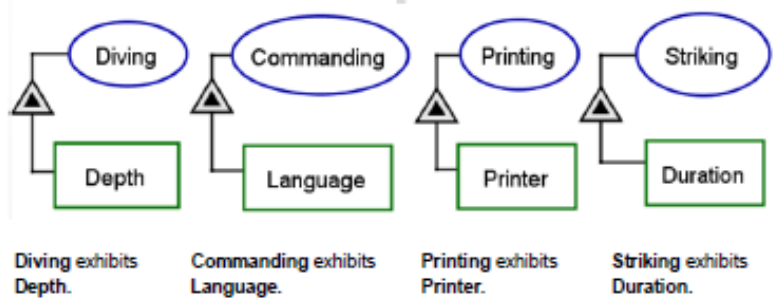
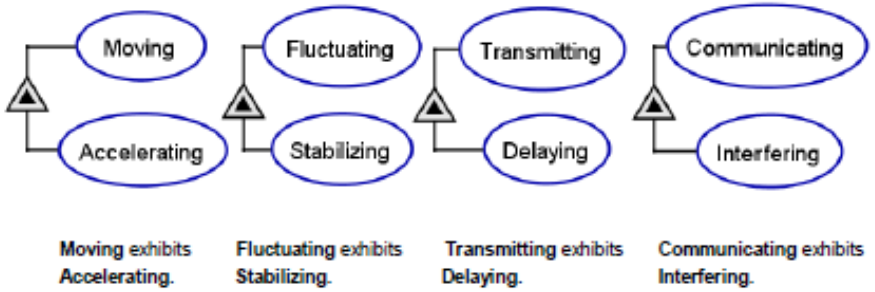
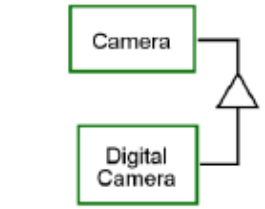


Figure 22 — Process exhibitor with attribute examples

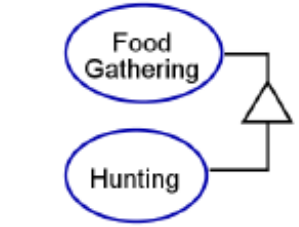


Generalization-Specialization

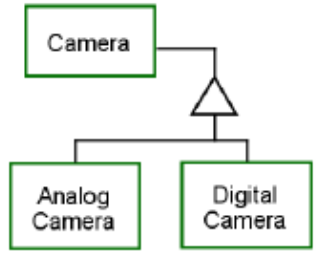
- A structural relation which denotes that one **thing** specializes to another **thing**.
- Commonly referred as inheritance in OO modelling languages.
  - For a complete collection of n specializations of a general that is an object, the syntax of the generalization-specialization relation link OPL sentence shall be: ***Specialization-object1, Specialization-object2, …,*** *and* ***pecialization-objectn are General-object*****.**
  - For a complete collection of n specializations of a general that is a process, the syntax of the generalization-specialization relation link OPL sentence shall be: ***Specialization-process1, Specialization-process2, …,*** *and* ***Specialization-processn are General-process*****.**



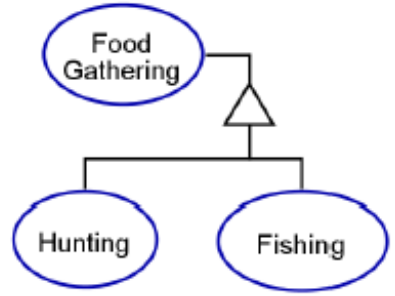
Digital Camera is a Camera.



Hunting is Food Gathering.



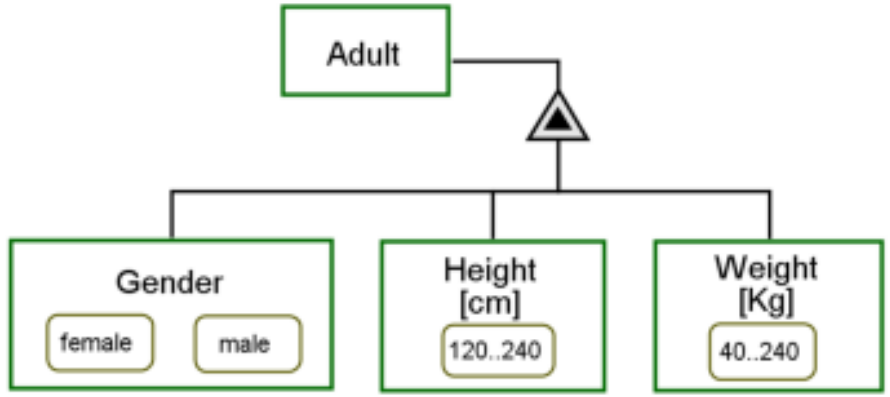
Analog Camera and Digital Camera are Cameras.



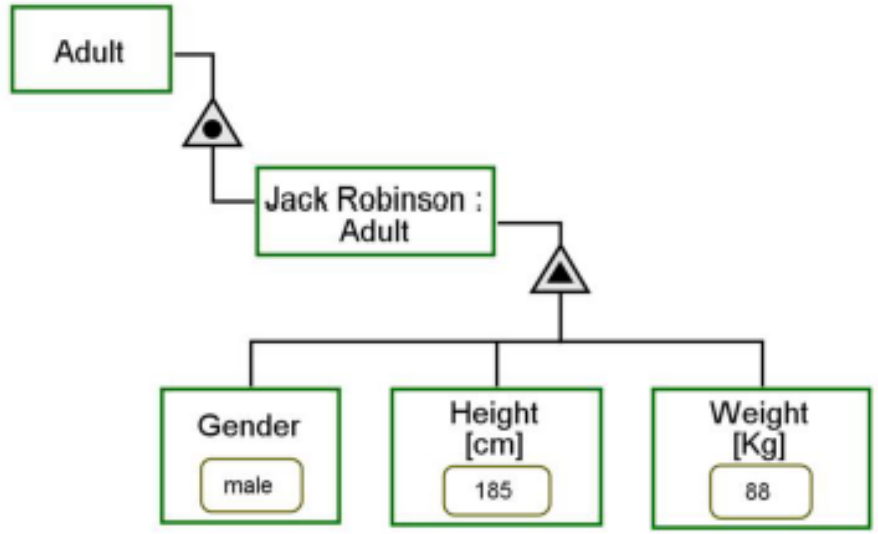
Hunting and Fishing are Food Gathering.

Classification-Instantiation

- The classification, which is an object class or a process class, is a source pattern for a thing connecting with one or more destination things, which are instances of the source thing's pattern, i.e. the qualities the pattern specifies acquire explicit values to instantiate the instance thing.
- An instance of a class shall be an incarnation of a particular identifiable instance of that class with the same classification identifier.
  - The syntax of the classification-instantiation relation link between an object class and a single instance shall be: ***Instance-object is an instance of Class-object*.**
  - The syntax of the classification-instantiation relation link between a process class and a single instance shall be: ***Instance-process is an instance of Class-process*.**
  - The syntax of the classification-instantiation relation link between a process class and n instances shall be; ***Instance-object1, Instance-object2, …, Instance-objectn are instances of Class-object*.**
  - The syntax of the classification-instantiation relation link between a process class and n instances shall be; ***Instance-process1, Instance-process2, …, Instance-processn are instances of Class-process*.**

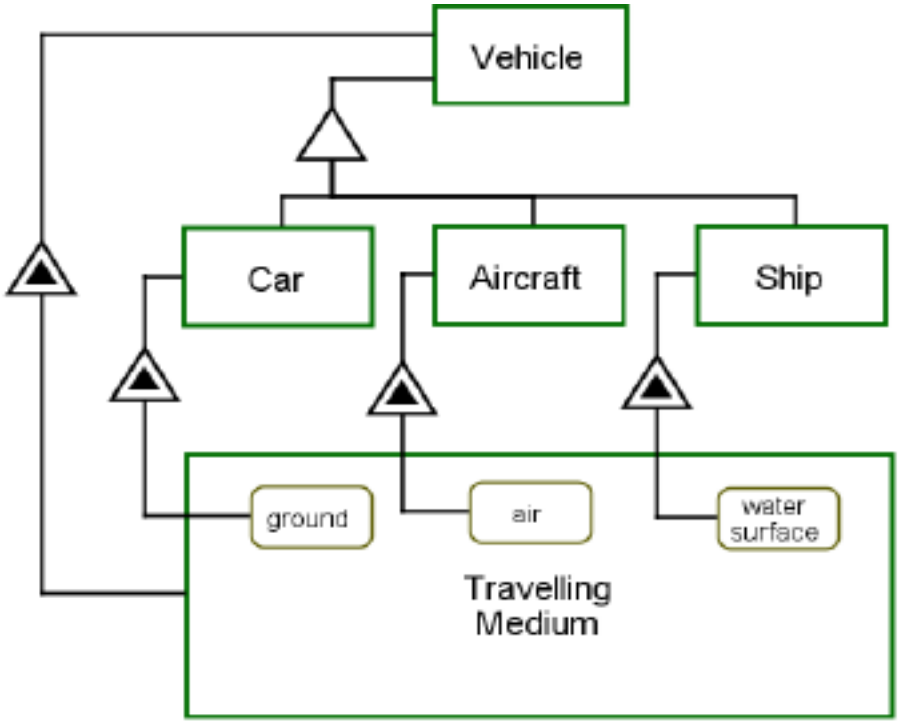


Adult exhibits **Gender**, **Height in cm**, and **Weight in Kg**.
Gender of **Adult** can be **female** or **male**.
Height in **cm** of **Adult** ranges from **120** to **240**.
Weight in **Kg** of **Adult** range from **40** to **240**.

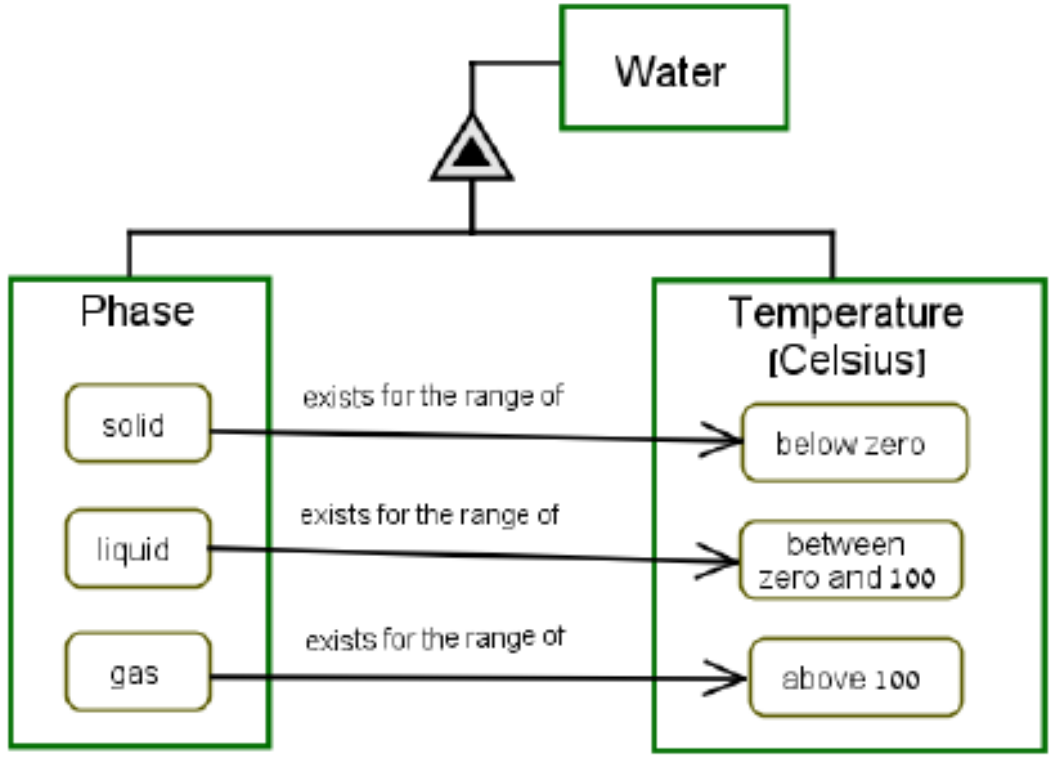


Jack Robinson is an instance of **Adult**.
Gender of **Jack Robinson** is **male**.
Height in **cm** of **Jack Robinson** is **185**.
Weight in **kg** of **Jack Robinson** is **88**.

States can be also used in some relations

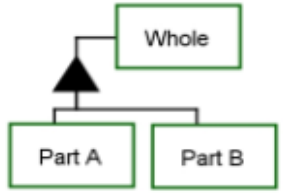
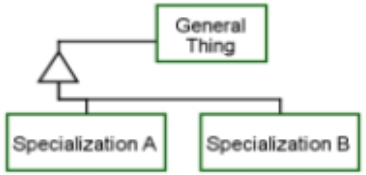
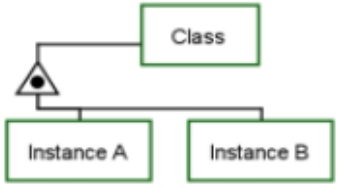
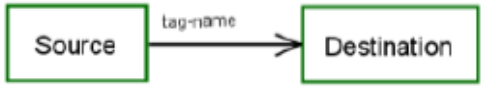
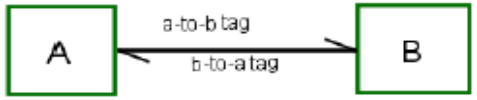
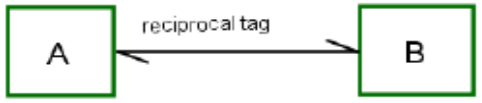


Vehicle exhibits **Travelling Medium.**
Travelling Medium of **Vehicle** can be **ground**, **air**, and **water surface.**
Car, **Aircraft**, and **Ship** are **Vehicles.**
Car exhibits **ground Travelling Medium.**
Aircraft exhibits **air Travelling Medium.**
Ship exhibits **water surface Travelling Medium.**



Water exhibits **Phase** and **Temperature** in **Celsius.**
Phase can be **solid**, **liquid**, or **gas.**
Temperature in **Celsius** can be **below zero**, **between zero and 100**, or **above 100.**
Solid Phase exists for the range of below zero Temperature in **Celsius.**
Liquid Phase exists for the range of between zero and 100 Temperature in **Celsius.**
Gas Phase exists for the range of above 100 Temperature in **Celsius.**

Summary

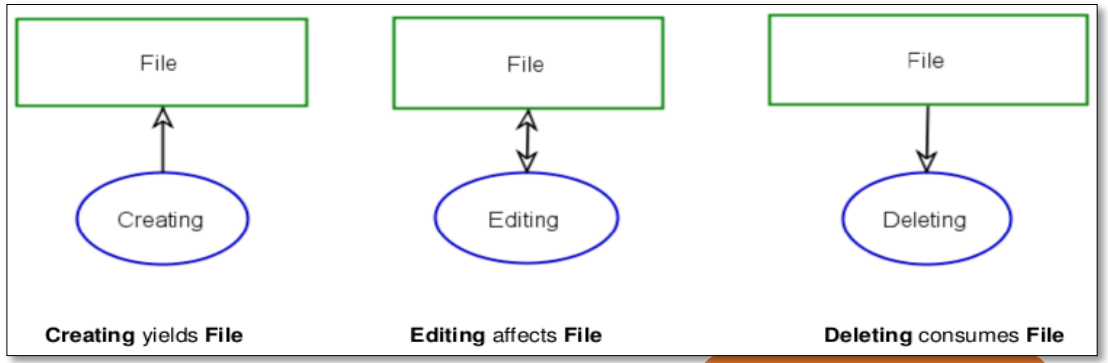
Aggregation- Participation		**Whole** consists of **Part A** and **Part B**.	–
Exhibition- **Characterization**		**Exhibitor** exhibits **Attribute A** as well as **Operation B**.	–
Generalization- Specialization		–	**Specialization A** and **Specialization B** are **General Thing.**
Classification- Instantiation		–	**Instance A** and **Instance B** are instances of **Class.**
Unidirectional tagged [Unidirectional null tagged]		**Source tag-name Destination.** [**Source** relates to **Destination.**]	
Bidirectional tagged		**A a-to-b tag B.** **B b-to-a tag A.**	
Reciprocal tagged [Reciprocal null tagged]		**A** and **B** are **reciprocal tag.** [**A** and **B** are related.]	

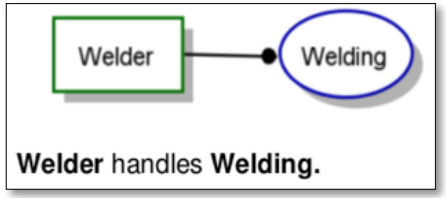
BEHAVIOR

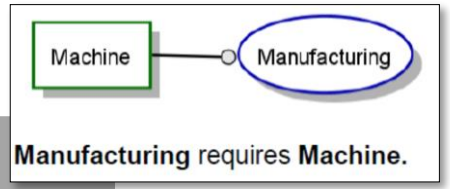
Procedural Links

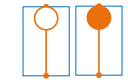
- **Procedural Link** is a link that specifies a **dynamic aspect** of the system by connecting na object (or one of its states) and a process.

Procedural Links









Enabling

- Agent
- Instrument

Transforming

- Consume
- Create
- Effect

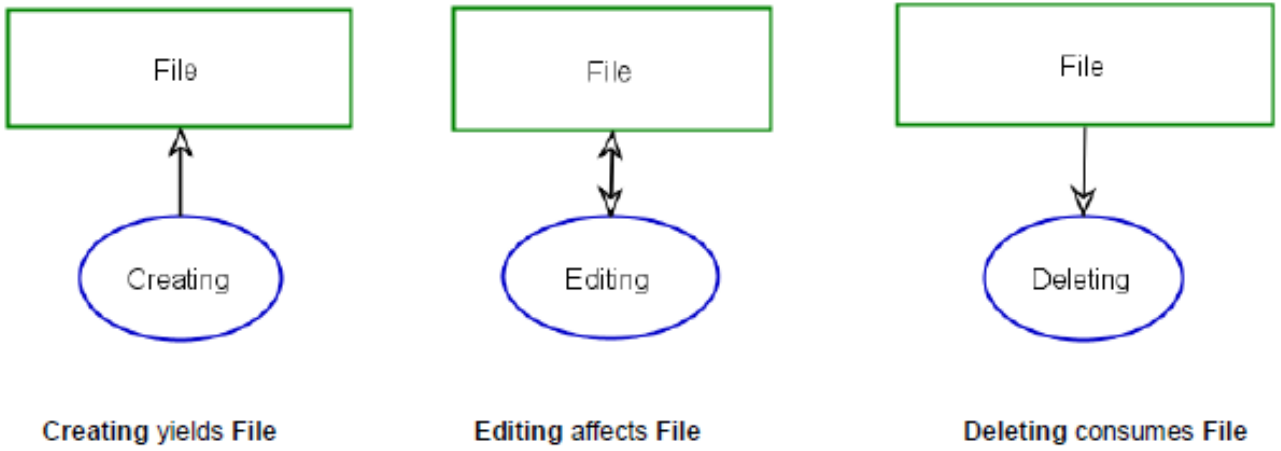
Invocation

Procedural

- Procedural links symbolize the behavior of the modeled system.
- Three types:
  - Enabling links: link a **process** to an **object** that enables the **process** but is not affected by it.
  - Transforming links: links a **process** to an **object** that is affected by the **process**.
  - Invocation links: shortcut notation between two consecutive **processes**.

Transforming Links

- A transformation link is a procedural link that connects a process with an object transformed by the process. Three types:
  - Consumption: the linked object is consumed and eliminated by the process.
  - Result: the linked object is constructed by the process.
  - Effect: the linked object is changed by the process.



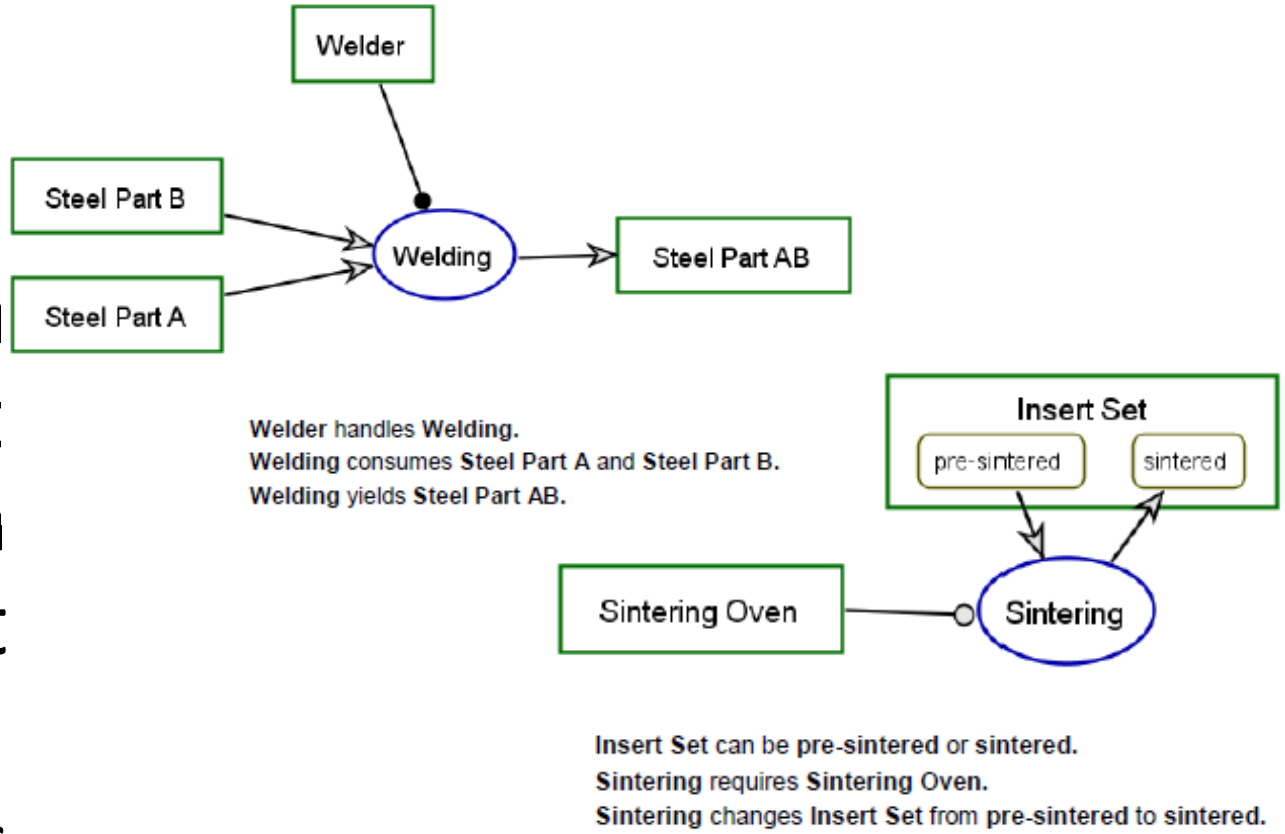
The syntax of a consumption link OPL sentence shall be: ***Processing consumes Consumee*.**

The syntax of a result link OPL sentence shall be: ***Processing yields Resultee*.**

The syntax of an effect link OPL sentence shall be: ***Processing affects Affectee*.**

Enabling Links

- An enabling link is a procedural link that connects a process with an enabler object of that process. Two types:
  - **Agent**: an enabler who is a human or a group of humans.
  - **Instrument**: a non-human enabler

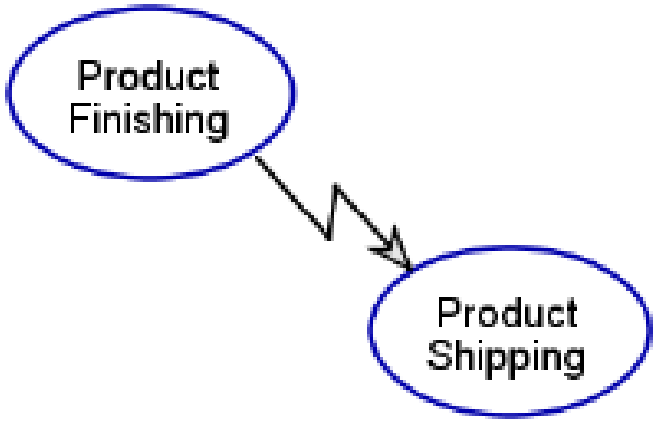


The syntax of an agent link OPL sentence shall be: ***Agent handles Processing*.**

The syntax of an instrument link OPL sentence shall be: ***Processing requires Instrument*.**

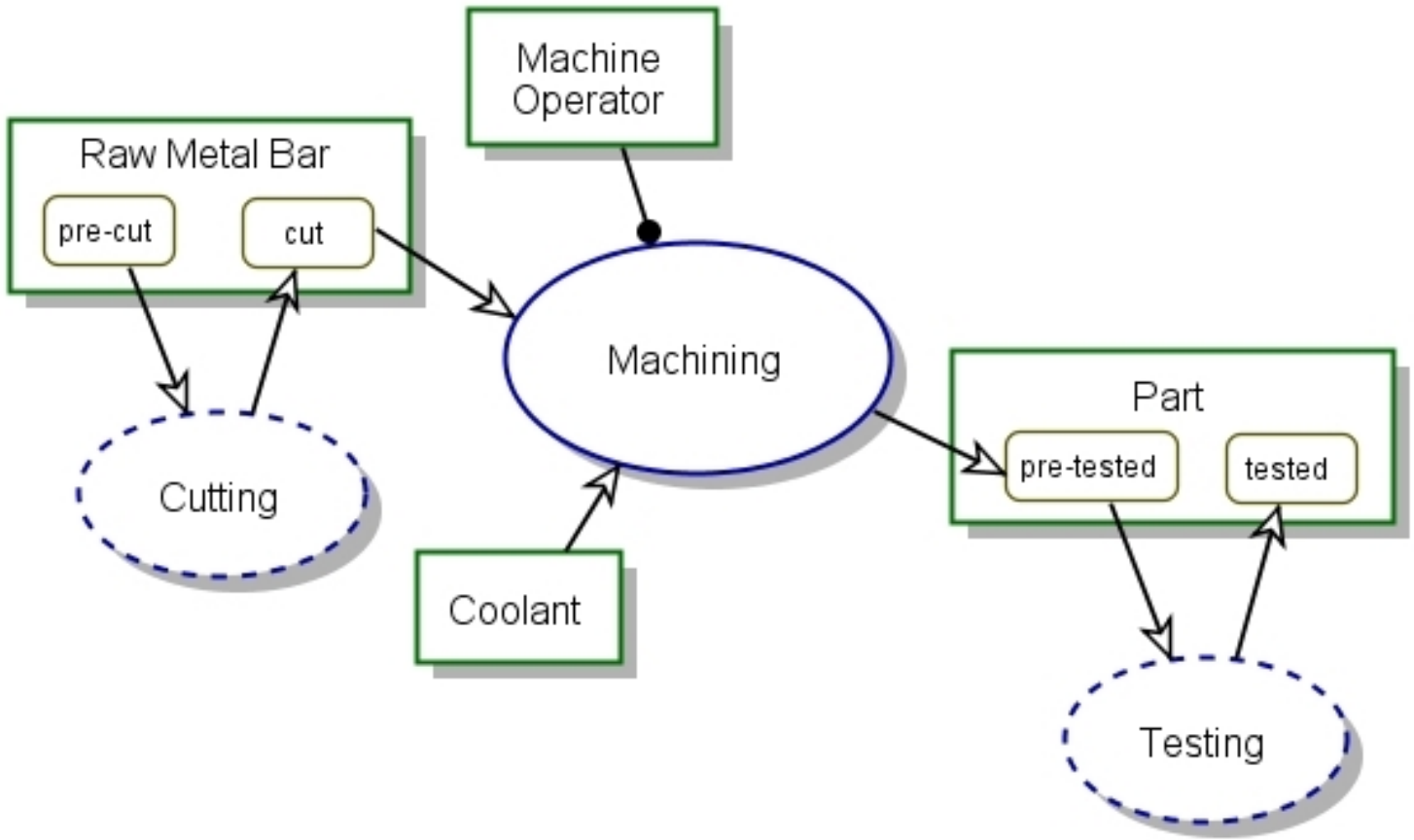
Invocation Link

- By definition, a process must transform an object. But sometimes the result of a process is not significant to the system and may be ignored. However, the result of the process is significant to a consecutive process.
- The invocation link provides a shortcut to bypass the modeling of the irrelevant object.
  - The syntax of an invocation link OPL sentence shall be: ***Invoking-process invokes invoked-process*.**
  - The syntax of a self-invocation link OPL sentence shall be: ***Invoking-process invokes itself*.**

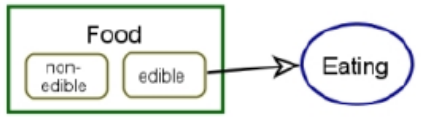
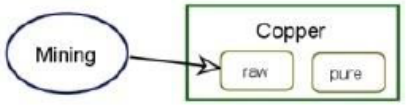
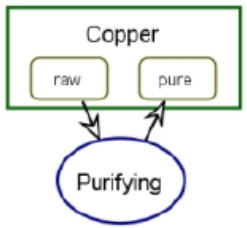
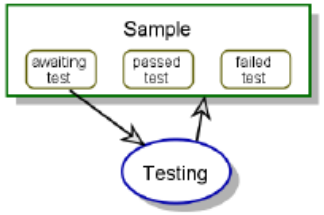
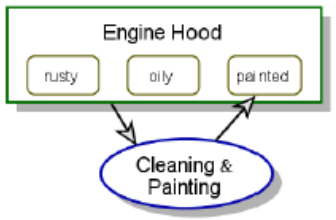


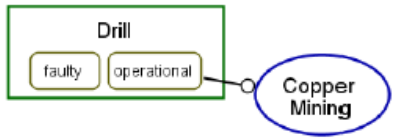
Product Finishing invokes **Product Shipping.**

States can be also used in some relations



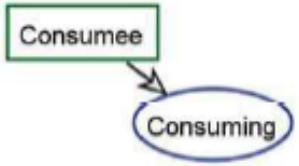
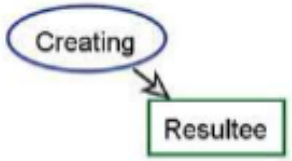
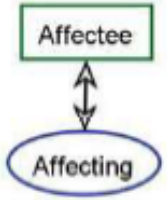
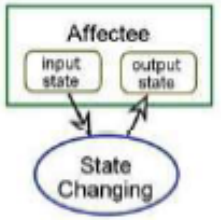
Raw Metal Bar is **physical.**
Raw Metal Bar can be **pre-cut** or **cut.**
Machine Operator is **physical.**
Coolant is **physical.**
Machining is **physical.**
Machining requires **Coolant.**
Machine Operator handles **Machining.**
Part is **physical.**
Part can be **pre-tested** or **tested.**
Testing is **environmental** and **physical.**
Cutting changes **Raw Metal Bar** from **pre-cut** to **cut.**
Machining consumes **Raw Metal Bar.**
Machining yields **pre-tested Part.**
Testing changes **Part** from **pre-tested** to **tested.**

Name	Semantics	Sample OPD & OPL	Source	Destination
State-specified consumption link	The process consumes the object if and only if the object is in the specified state.	 **Eating** consumes **edible Food.**	consumee state	process
State-specified result link	The process generates the object in the specified state.	 **Mining** yields **raw Copper.**	process	resultee state
Input-output-specified effect link pair (consisting of one state-specified *input link* and one state-specified *output link*)	The process changes the object from a specified input state via the *input link* to a specified output state via the *output link*.	 **Purifying** changes **Copper** from **raw** to **pure.**	affectee source state	affecting process
			affecting process	affectee destination state
Input-specified effect link pair (consisting of one state-specified *input link* and one state-unspecified *output link*)	The process changes the object from a specified input state to any output state.	 **Testing** changes **Sample** from **awaiting test.**	affectee source state	affecting process
			affecting process	affectee
Output-specified effect link pair (consisting of one state-unspecified *input link* and one state-specified *output link*)	The process changes the object from any input state to a specified output state.	 **Cleaning & Painting** changes **Engine Hood** to **painted.**	affectee	affecting process
			affecting process	affectee destination state

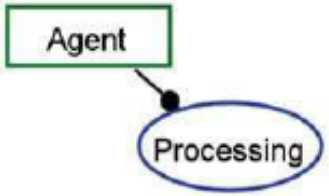
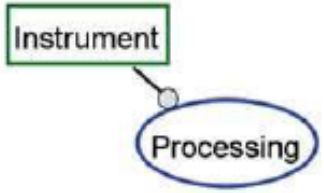
Name	Semantics	Sample OPD & OPL	Source	Destination
State-specified agent link	The human agent enables the process provided she is at the specified state.	 **Healthy Miner** handles **Copper Mining.**	agent state	enabled process
State-specified instrument link	The process requires the instrument at the specified state.	 **Copper Mining** requires **operational Drill.**	instrument state	enabled process

Summary

Procedural transforming links

consumption link	result link	effect link	in-out link pair
			
Consuming consumes **Consumee.**	**Creating** yields **Resultee.**	**Affecting** affects **Affectee.**	**State Changing** changes **Affectee** from **input state** to **output state.**

Procedural enabling links

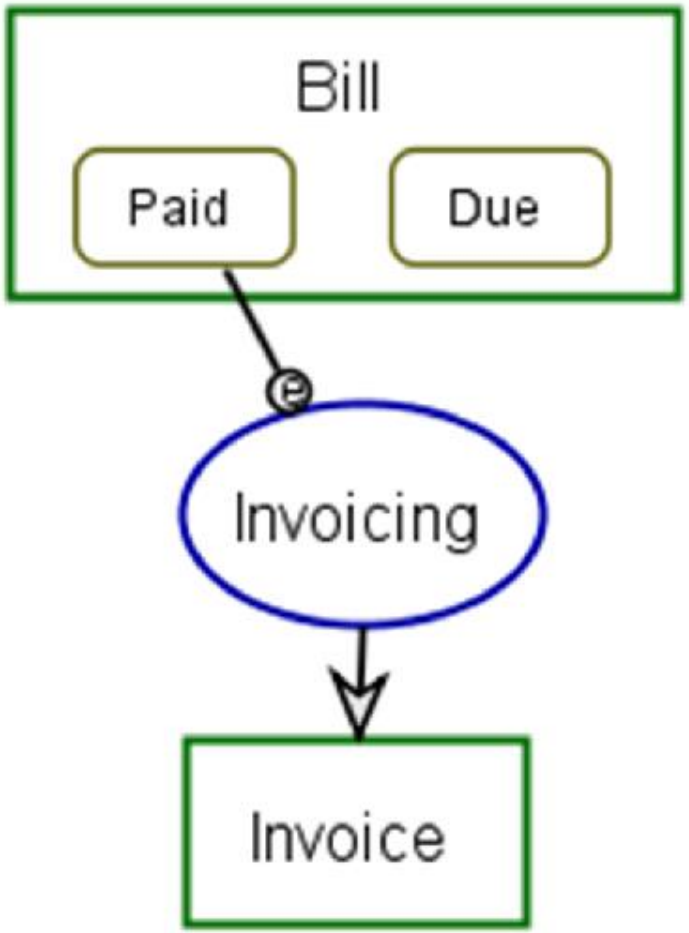
agent link	instrument link
	
Agent handles **Processing.**	**Processing** requires **Instrument.**

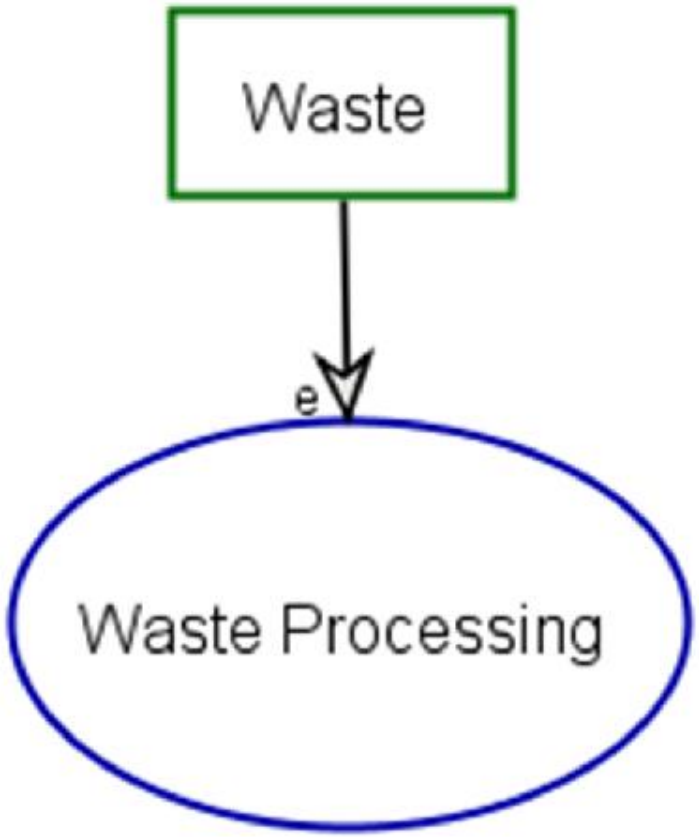
CONDITIONS & EVENTS

Advanced Procedural Links

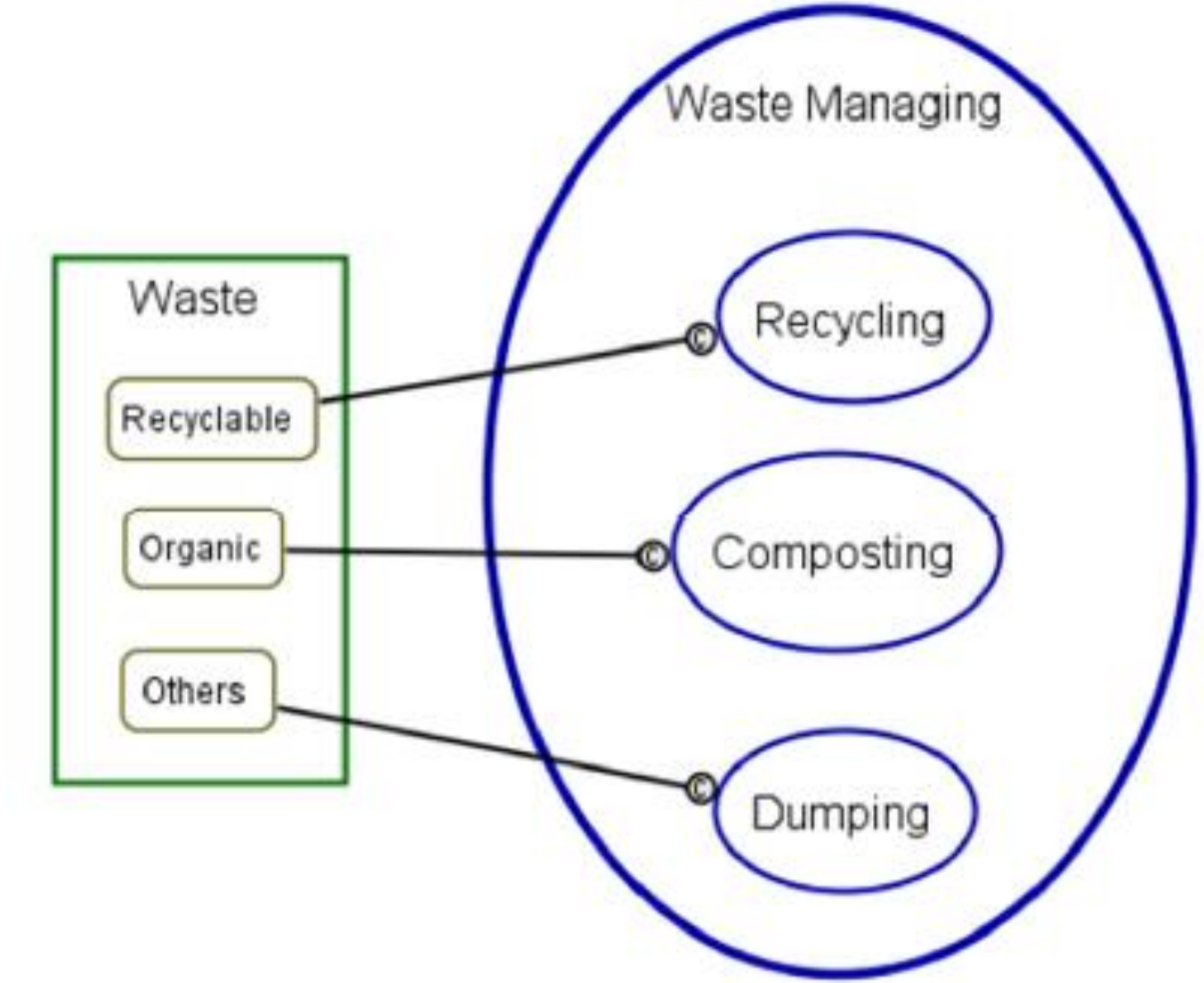
- **Event links**: triggers process activation if the event is satisfied. They are **used when agents are not controlling the process**. Two kinds of event links:
  - **Instrument event link**: the process is triggered if the **instrument object exists** (or is in a specific state).
  - **Consumption event link**: the process is triggered if the **consumed object exists** (or is in a specific state). The object is then consumed.
- **Condition link**: conditions the execution of a process to the **existence of an object or to the object being in a specific state**. If the condition is not matched, the process is skipped.

Event Link





Conditional Link

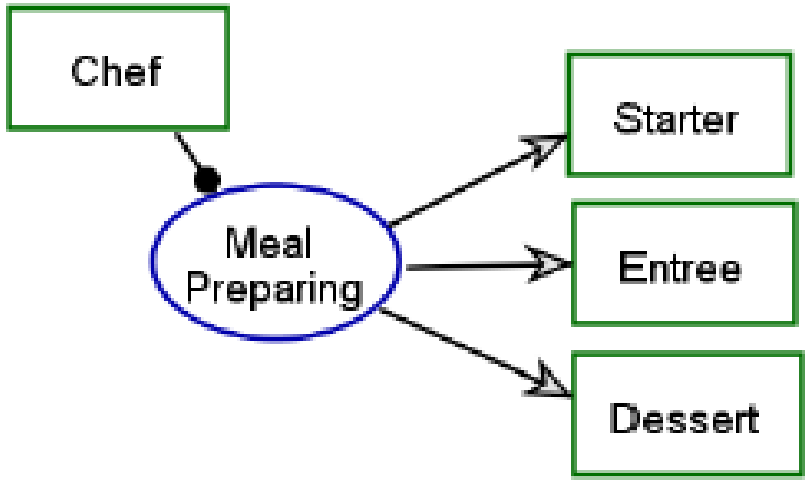


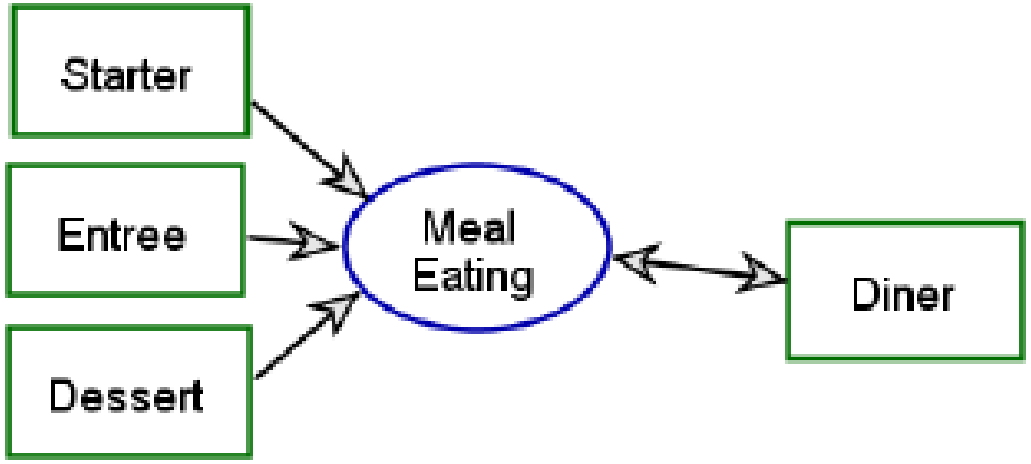
LOGICAL OPERATIONS

AND, XOR, and OR

- A group of two or more procedural links of the same kind that originate from, or arrive at, the same process shall have the semantics of logical **AND**.
- A group of two or more procedural links of the same kind that **originate from a common point**, or arrive at a common point, on the same object or process shall be a link fan. A link fan shall follow the semantics of either a **XOR or an OR** operator.
  - The **XOR operator** shall mean that **exactly one** of the things at the divergent link end of the link fan exists.
  - The **OR operator** shall mean that **at least one** of the two or more things at the divergent end of the link fan exists.

AND

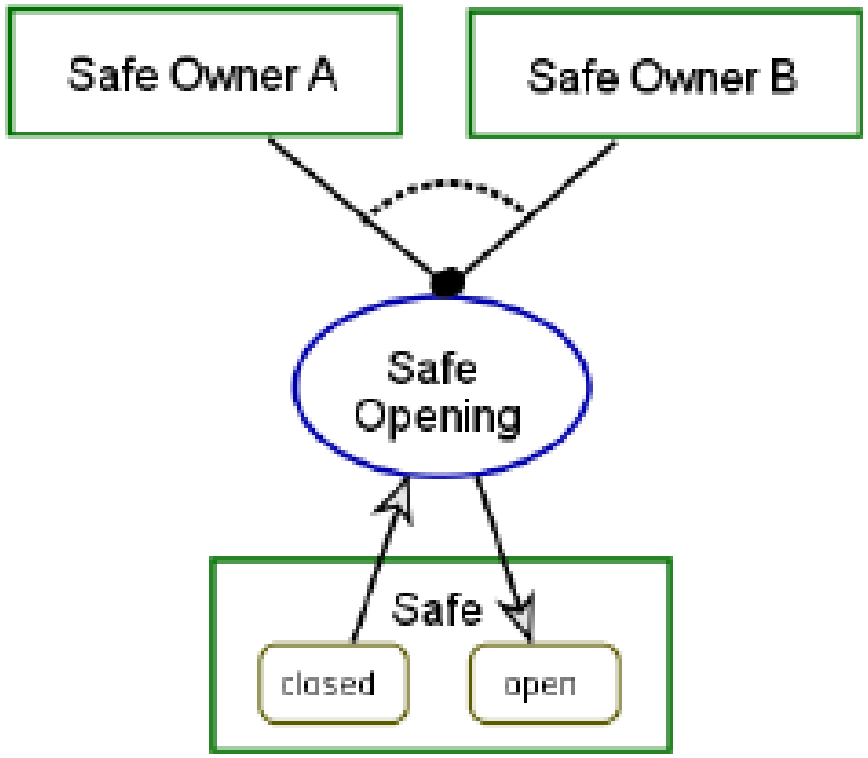




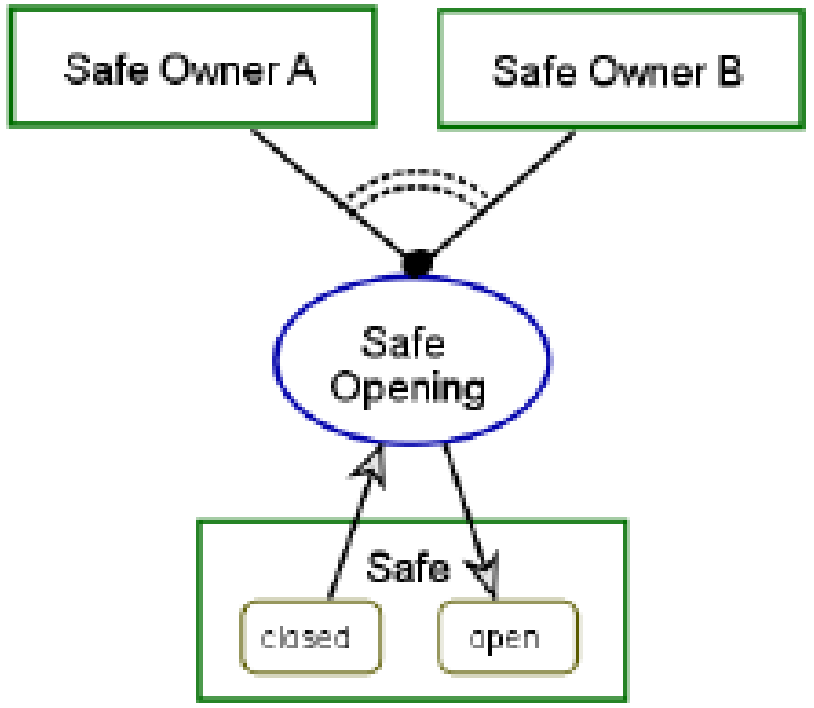
Chef handles **Meal Preparing**.
Meal Preparing yields **Starter**, Entree, and **Dessert**.

Meal Eating affects Diner.
Meal Eating consumes **Dessert**, Entree, and **Starter**.

XOR / OR



Exactly one of **Safe Owner A** and **Safe Owner B** handles **Safe Opening.**



At least one of **Safe Owner A** and **Safe Owner B** handles **Safe Opening.**

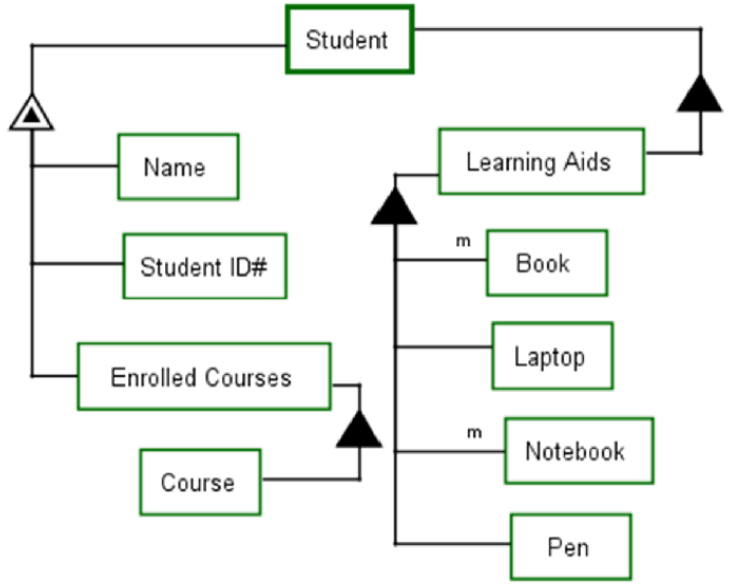
COMPLEXITY

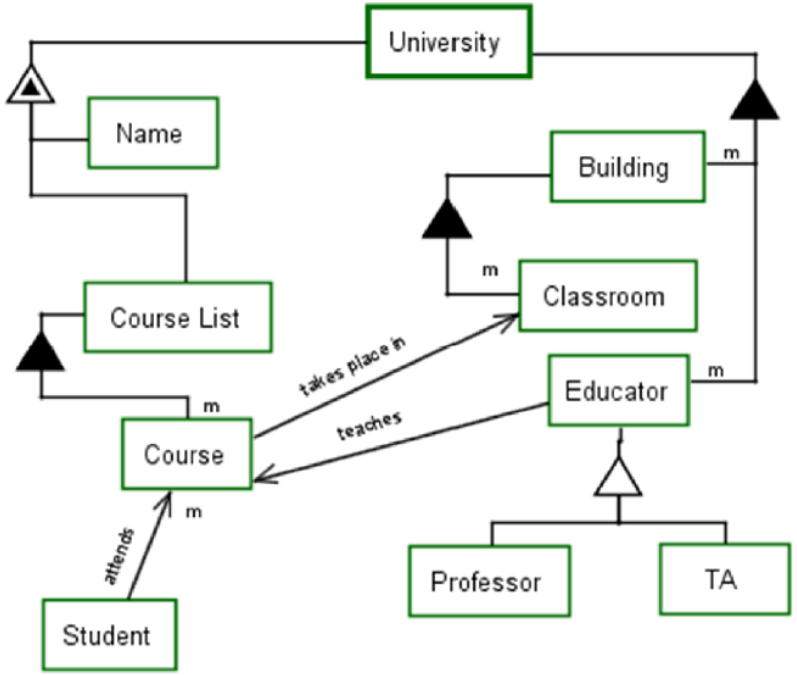
Managing Complexity

- OPM helps manage the system's complexity with three refinement mechanisms:
  - In-Zooming/Out-Zooming: primary used to model **process** flow.
  - Un-Folding/Folding: primary used to model **object** structure.
  - State Expression/Suppression: show only the **states** of an **object** that are relevant in the current context of the model.

Unfolding

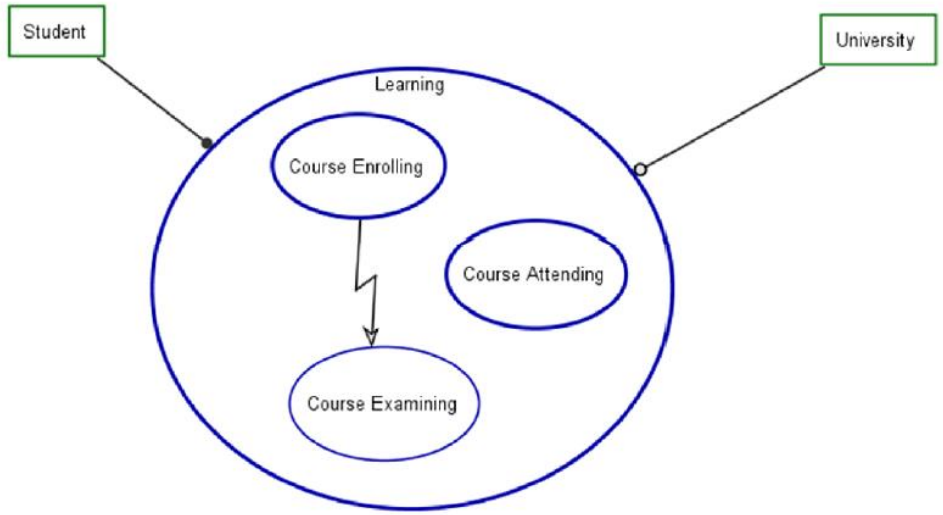
- Creates a new diagram with the unfolded object at the top.
- The unfolded object can now be refined by adding its parts and attributes.
- The attributes and parts can be further refined in the same diagram or with further unfolding

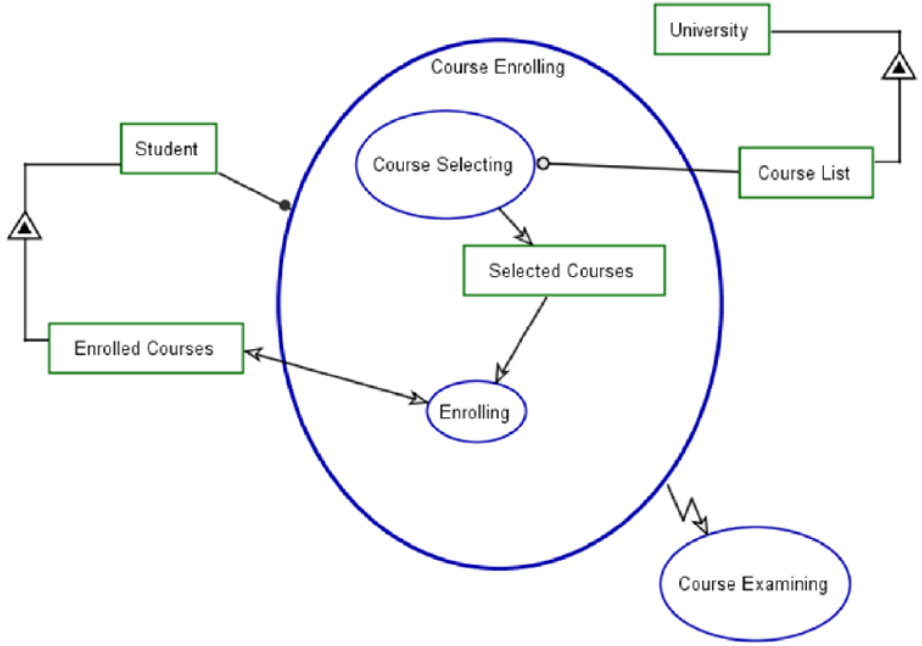


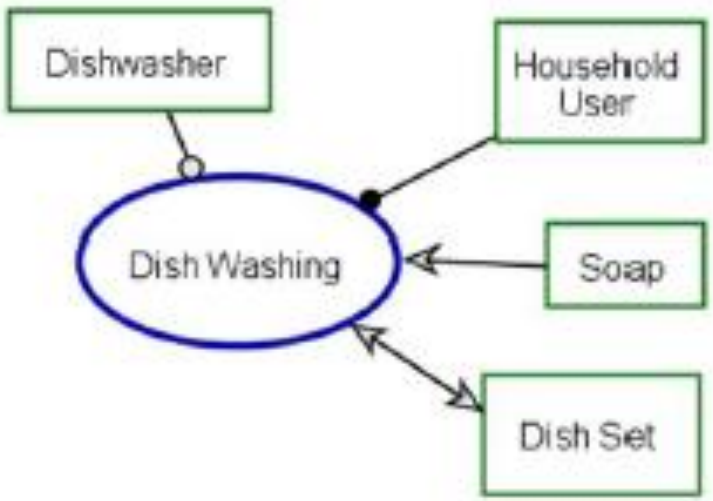


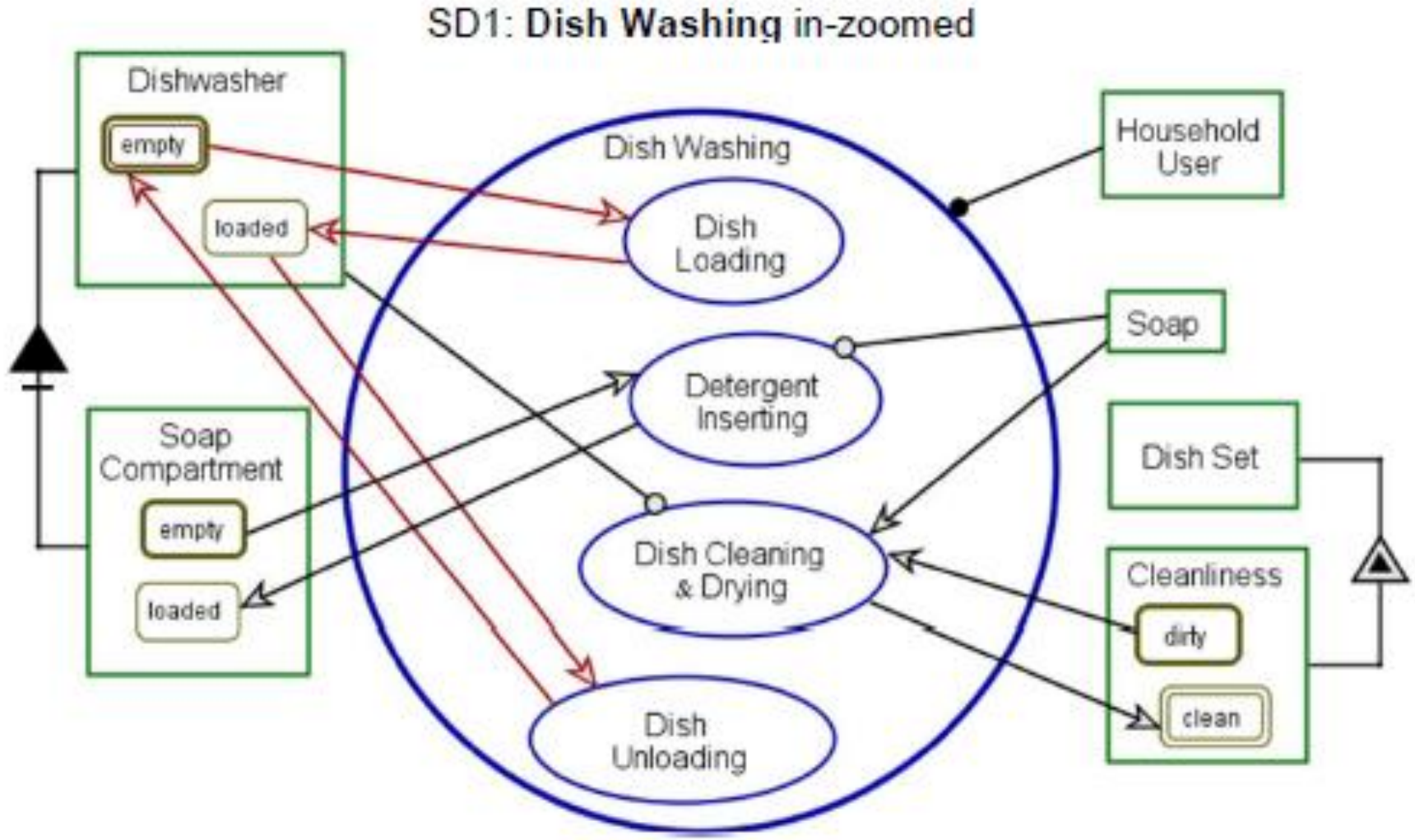
In-Zooming

- Creates a new diagram with the in-zoomed process centered and enlarged.
- The in-zoomed can now be refined by displaying the internal processes that compose it.
- The objects that are linked to the process can also be refined and connected to the processes that compose the in-zoomed process.
- The inner processes can be refined further by inzooming or unfolding.



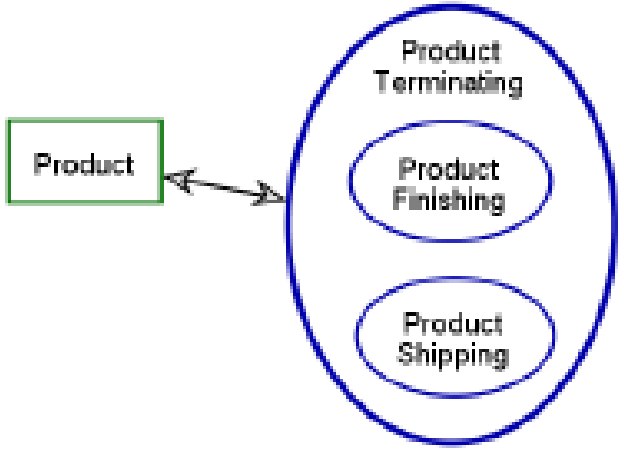



SD: Dish Washing System
Dishwasher
Household User
Dish Washing
Soap
Dish Set

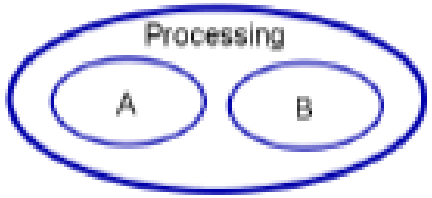

SD1: Dish Washing in-zoomed
Dishwasher
empty
loaded
Soap Compartment
empty
loaded
Dish Washing
Dish Loading
Detergent Inserting
Dish Cleaning & Drying
Dish Unloading
Household User
Soap
Dish Set
Cleanliness
dirty
clean

Execution Order

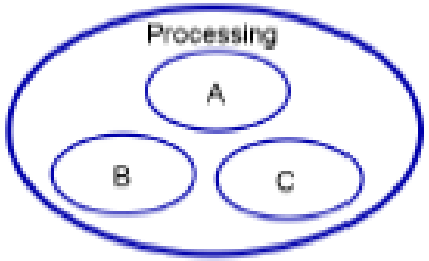
- An in-zoomed process can contain many subprocesses.
- When the flow of information in the model does not state otherwise, the processes inside an in-zoomed process are executed from top to bottom.
  - Upon subprocess completion within the context of an in-zoomed process, the subprocess immediately invokes the one(s) below it.
  - Top: Subprocesses A and B initiate in parallel as soon as Processing starts.
  - Bottom: Subprocesses B and C initiate in parallel as soon as subprocess A ends.



Product Terminating zooms into Product Finishing and Product Shipping, in that sequence.



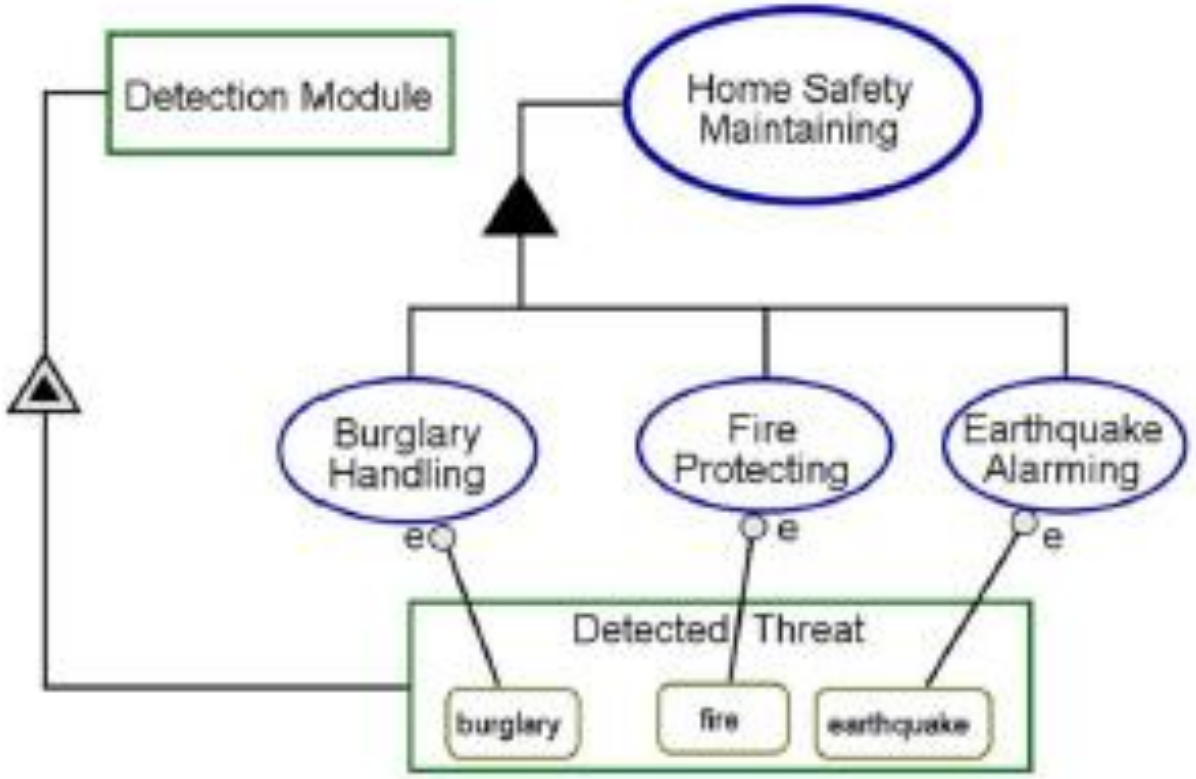
Processing zooms into parallel A and B.



Processing zooms into A and parallel B and C, in that sequence.

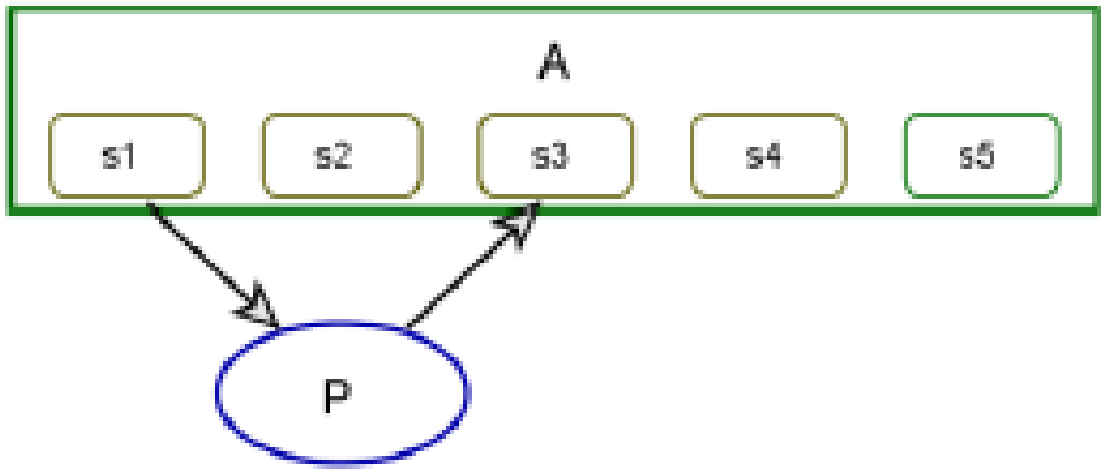
Asynchronous

The modelling of asynchronous process refinement shall use the aggregation-participation fundamental structural link either through in-diagram aggregation unfolding or as a new-diagram aggregation unfolding of the process.

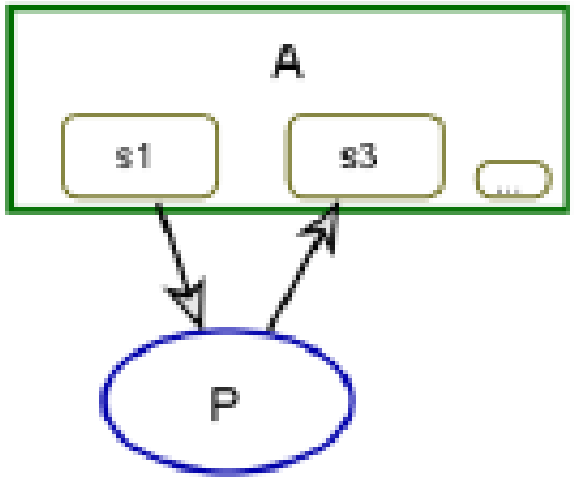


Home Safety Maintaining consists of **Burglary Handling**, **Fire Protecting**, and **Earthquake Alarming**.
Detection Module exhibits **Detection Treat**.
Detection Treat can be **burglary**, **fire**, or **earthquake**.
Burglary Detected Threat initiates **Burglary Handling**, which requires **burglary Detected Threat**.
Fire Detected Threat initiates **Fire Protecting**, which requires **fire Detected Threat**.
Earthquake Detected Threat initiates **Earthquake Alarming**, which requires **earthquake Detected Threat**

State Suppressing



A can be s1, s2, s3, s4, or s5.
P changes A from s1 to s3.

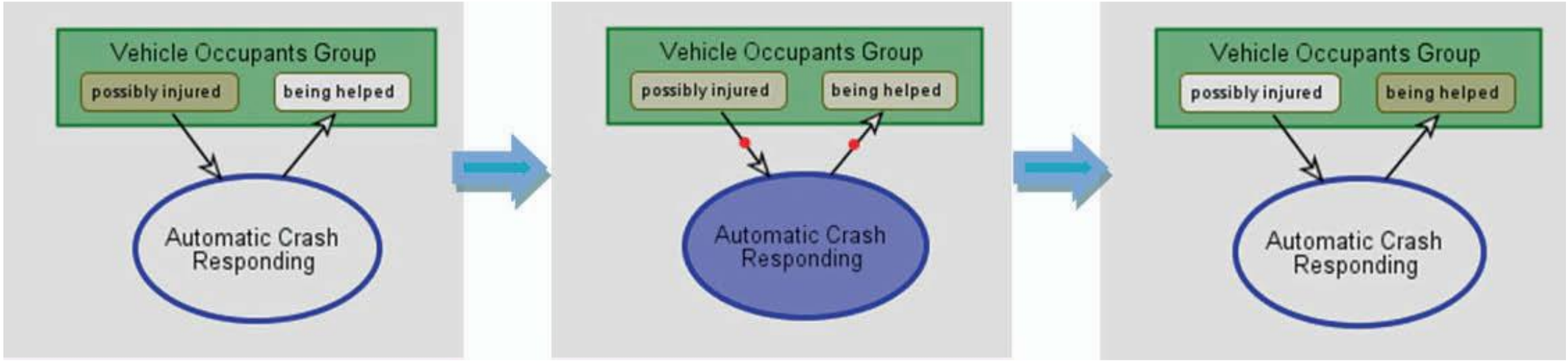


A can be s1, s3, or other states.
P changes A from s1 to s3.

SIMULATIONS

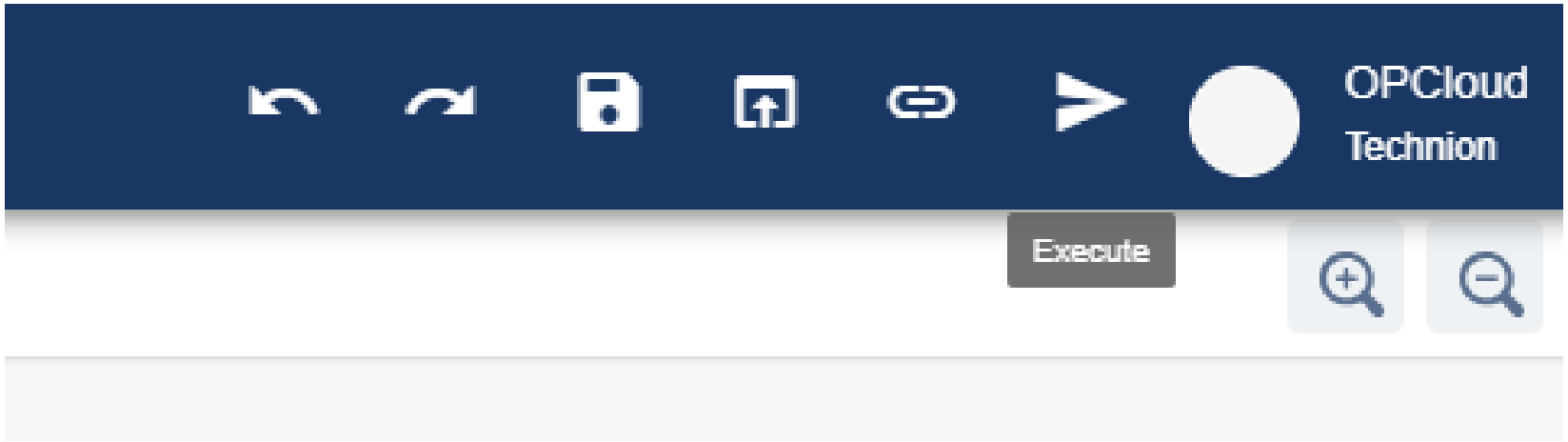
Execution

- One of the most attractive and useful features of an OPM model, which enables it to be visualized and tested, is its executability; that is, the ability to simulate a system by executing its model via animation in a properly designed software environment.



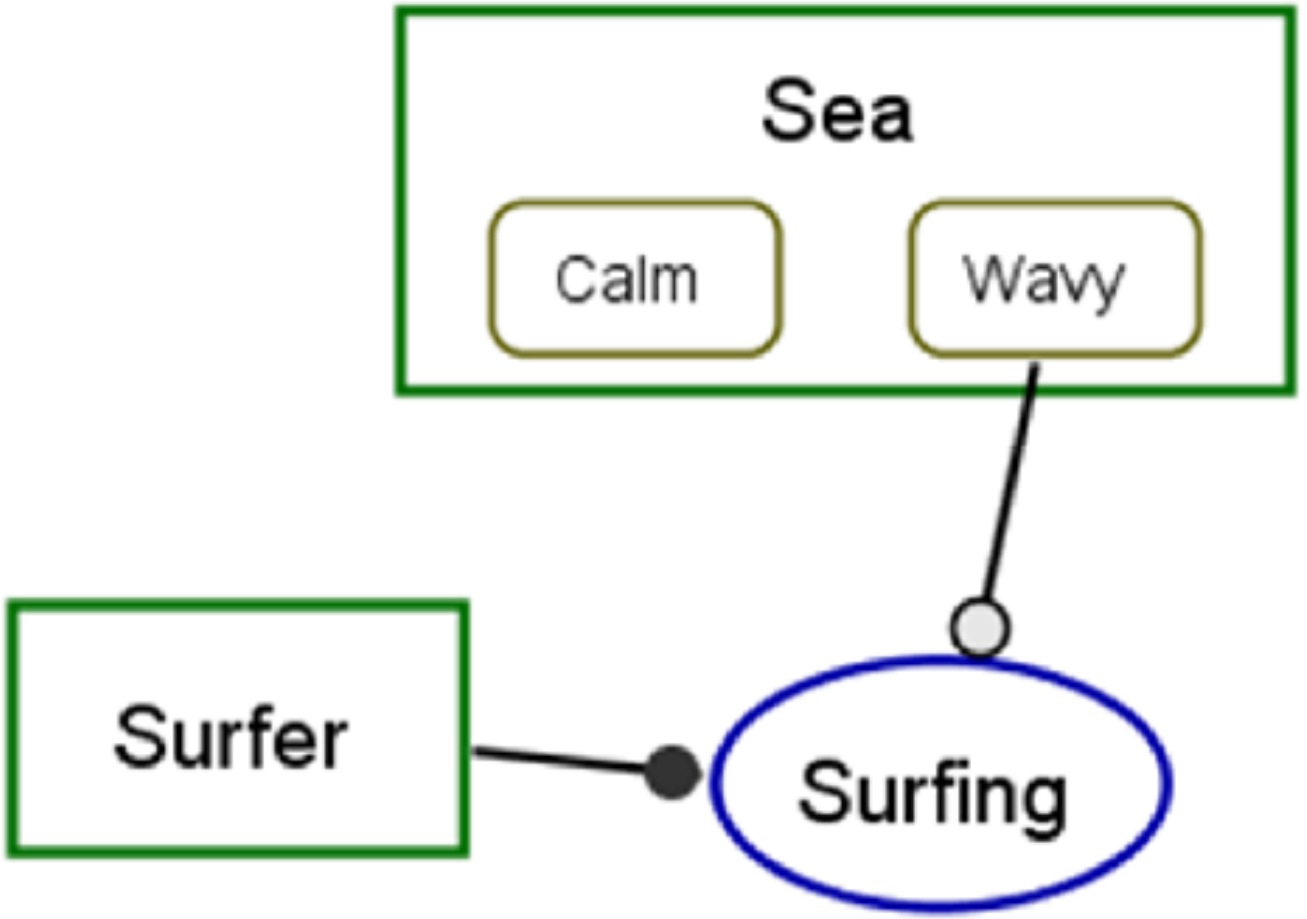
Execution

- To execute the OPM model, click the Execute button. To animate the execution, tokens, shaped as green circles, run along the links, following the operational semantics of OPM.
- If a process is computational, then once the result is calculated, a value slot is added to the relevant object(s) with the computed value recorded in it. In any case of missing values alert will be shown to the user with the description of the problem.

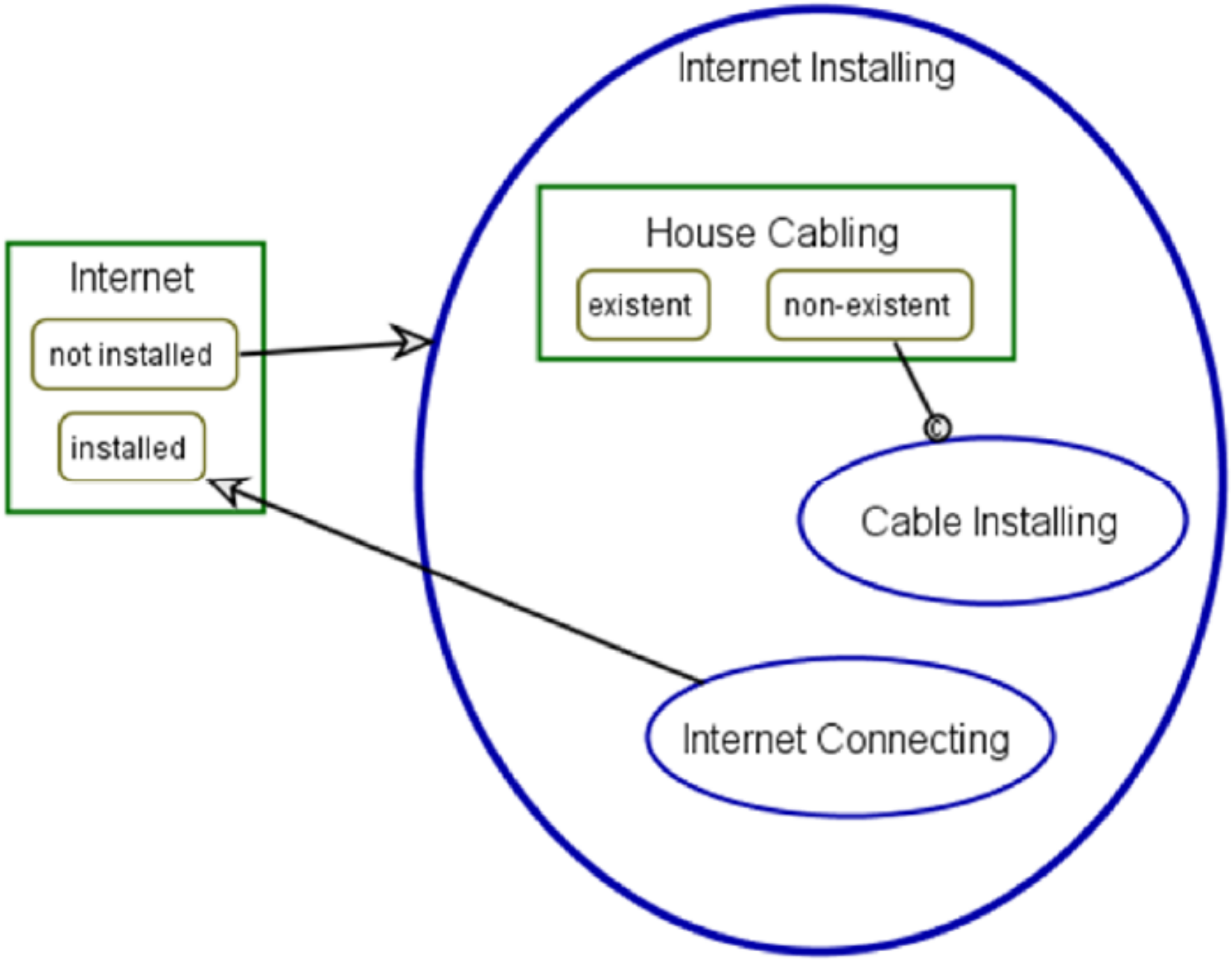


COMMON STRUCTURES

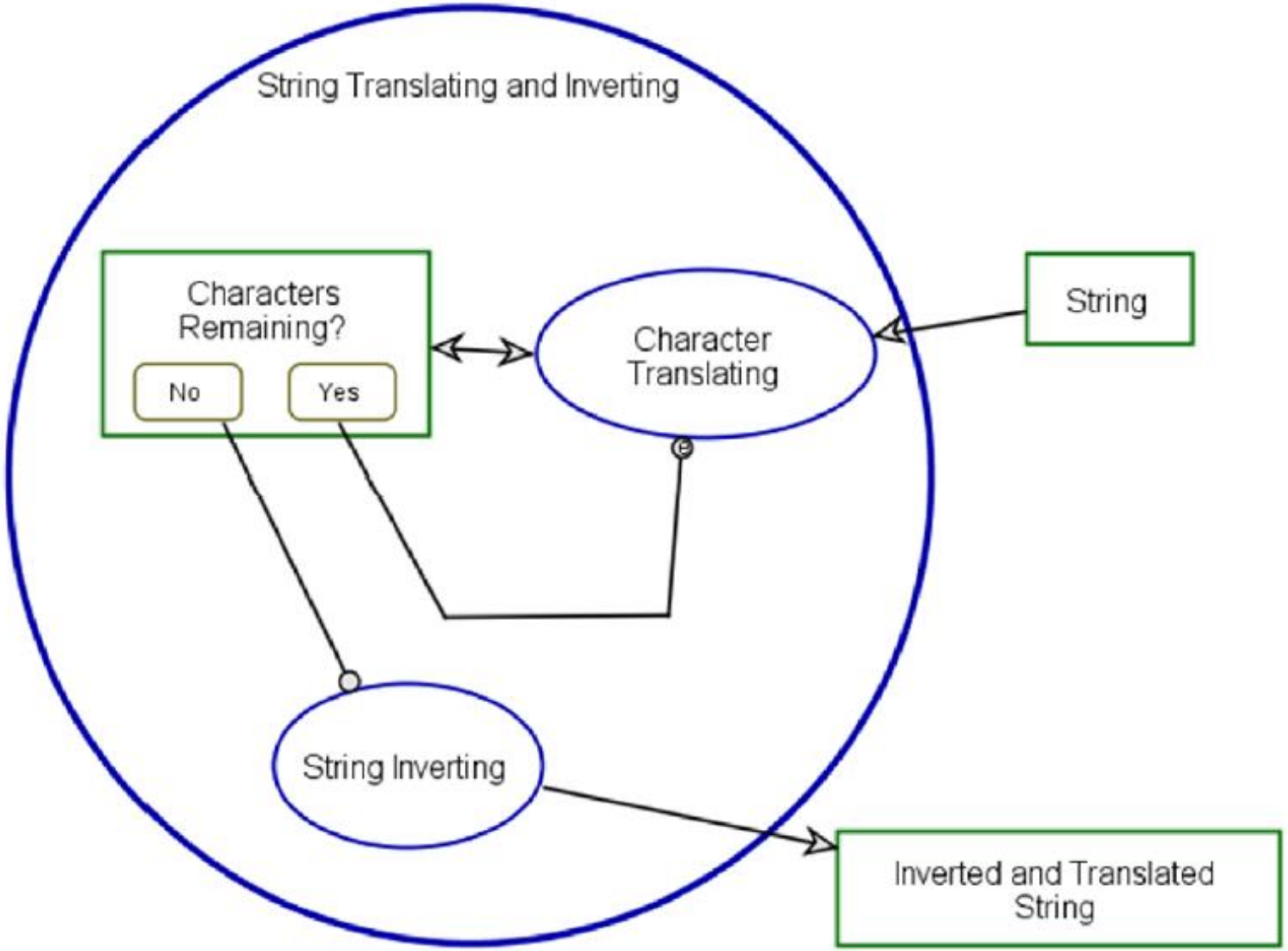
Wait until



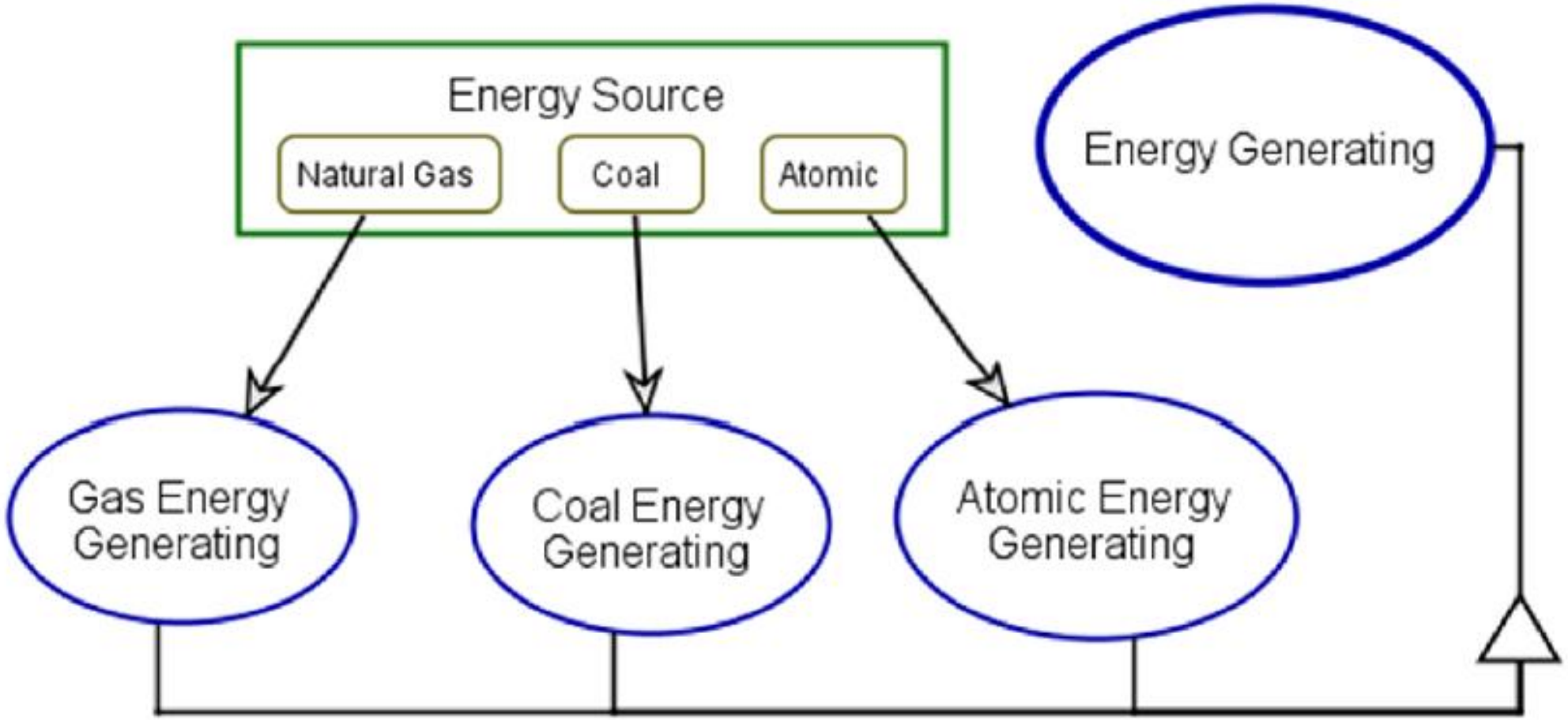
Do If


Internet Installing
House Cabling
existent
non-existent
Internet
not installed
installed
Cable Installing
Internet Connecting

Do while


String Translating and Inverting
Characters Remaining?
No
Yes
Character Translating
String
String Inverting
Inverted and Translated String

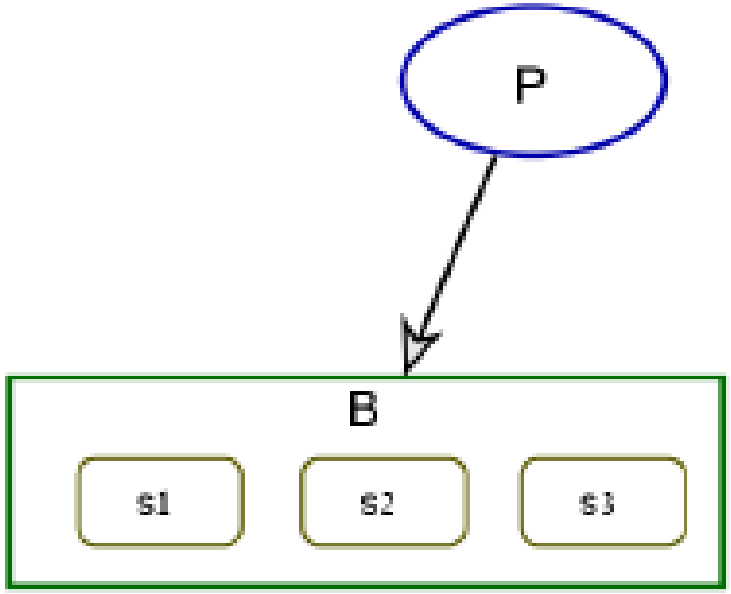
Case


Energy Source
Natural Gas
Coal
Atomic
Energy Generating
Gas Energy Generating
Coal Energy Generating
Atomic Energy Generating

END OF CLASS

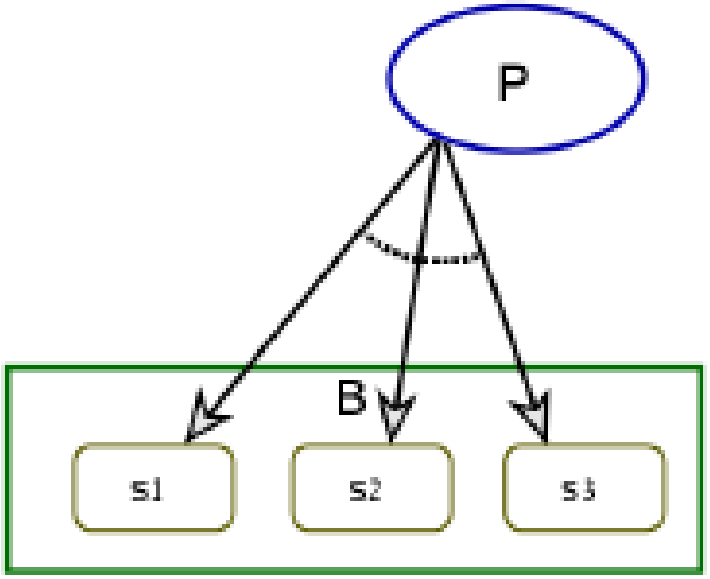
OTHER MECHANISMS

EQUIVALENT CHANCES



B can be **s1**, **s2**, or **s3**.

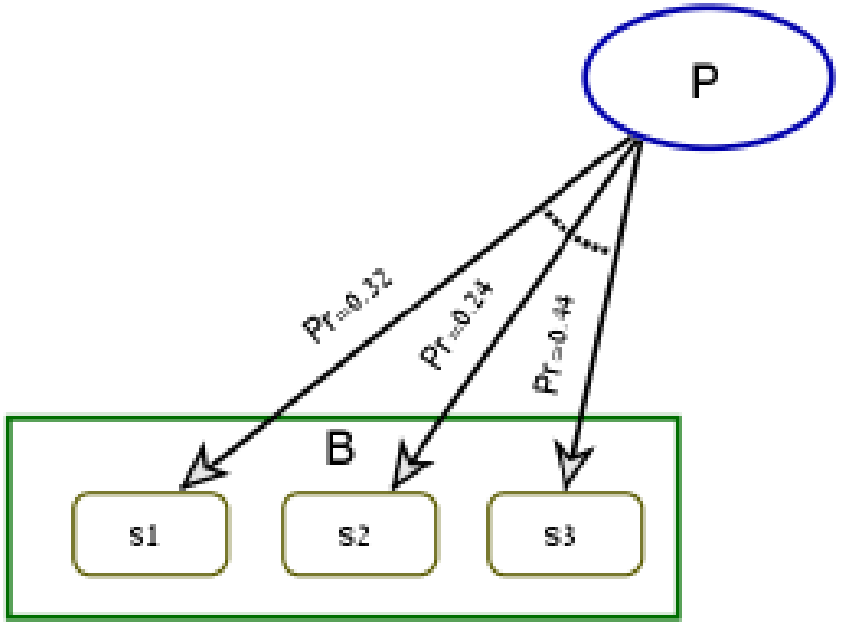
P yields B.



B can be **s1**, **s2**, or **s3**.

P yields exactly one of **s1** B, **s2** B, or **s3** B.

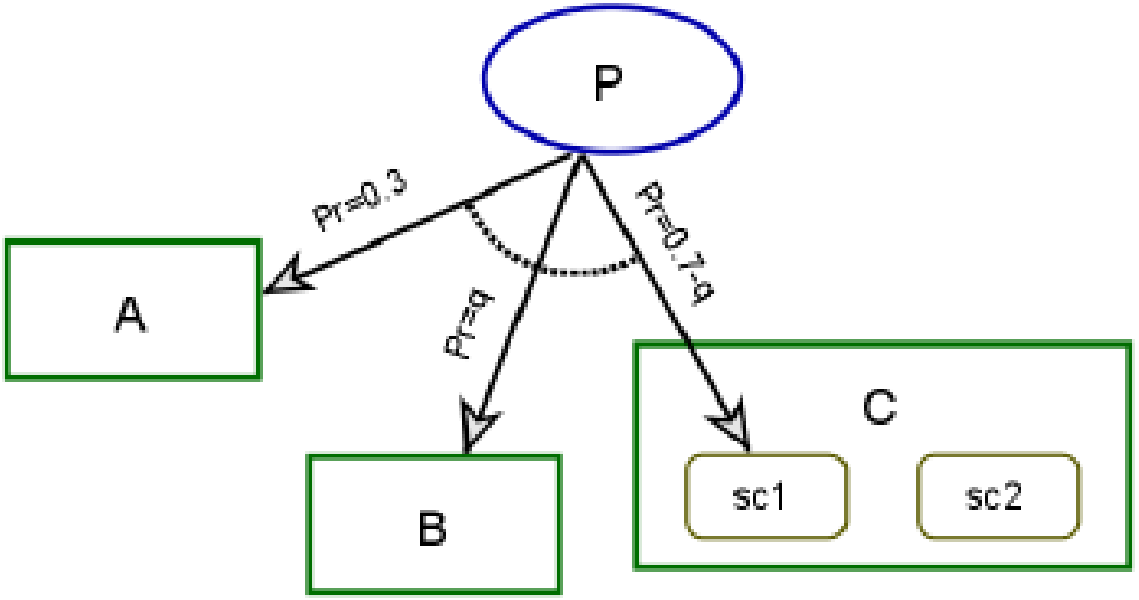
ADD PROBABILISTIC CHANCES



P yields **s1** B with probability **0.32**, **s2** B with probability **0.24**, or **s3** B with probability **0.44**.

The analogous deterministic case:

P yields exactly one of **s1** B, **s2** B, or **s3** B.

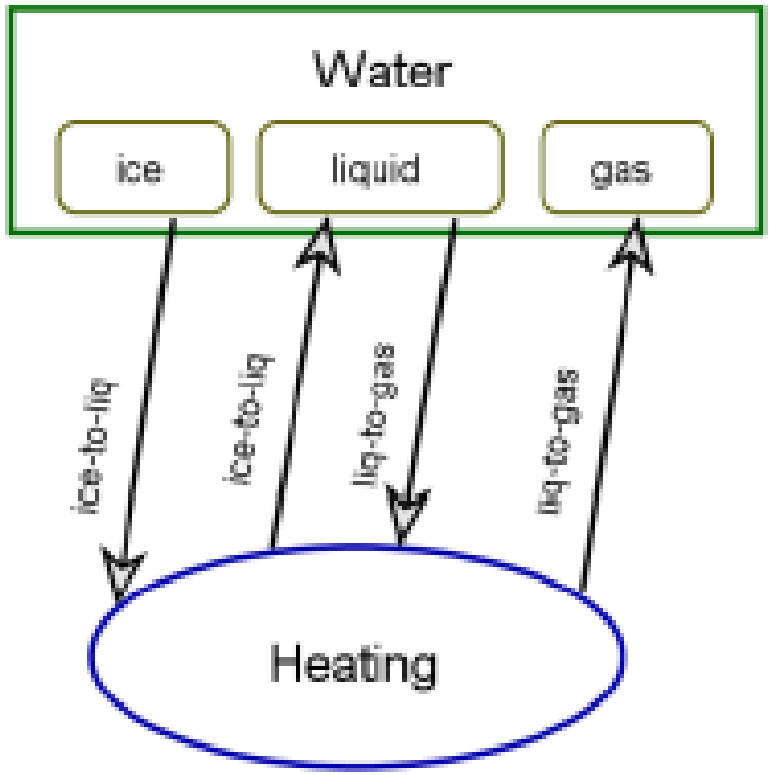


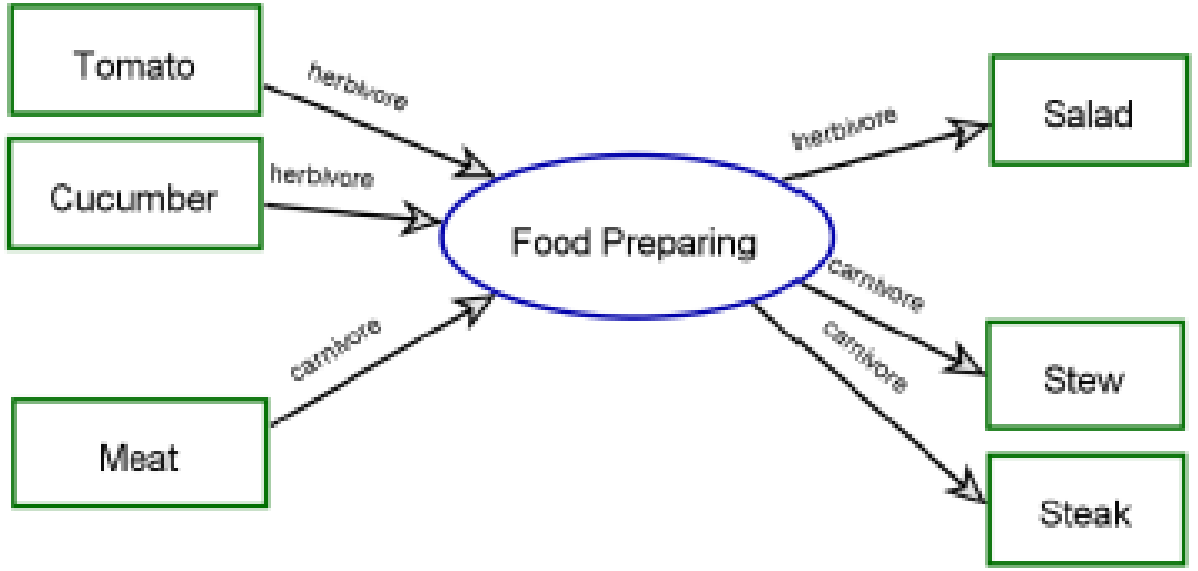
P yields **A** with probability **0.3**, **B** with probability **q**, or **sc1** C with probability **0.7-q**.

The analogous deterministic case:

P yields exactly one of **A**, **B**, or **sc1** C.

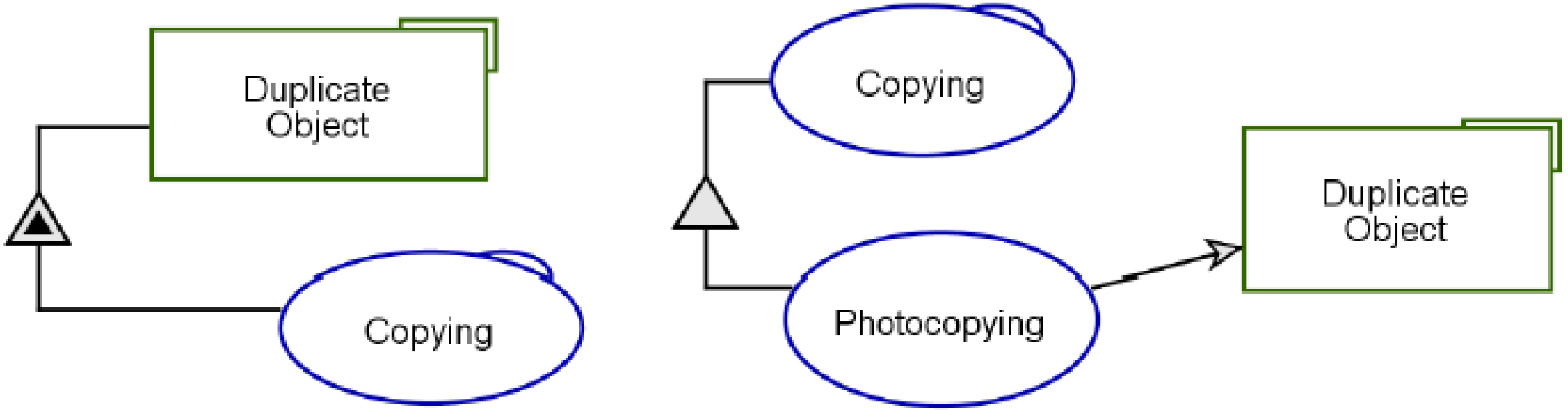
PATH LABELS





Following path **carnivore**, **Food Preparing** consumes **Meat.**
Following path **herbivore**, **Food Preparing** consumes **Cucumber** and **Tomato.**
Following path **carnivore**, **Food Preparing** yields **Stew** and **Steak.**
Following path **herbivore**, **Food Preparing** yields **Salad.**

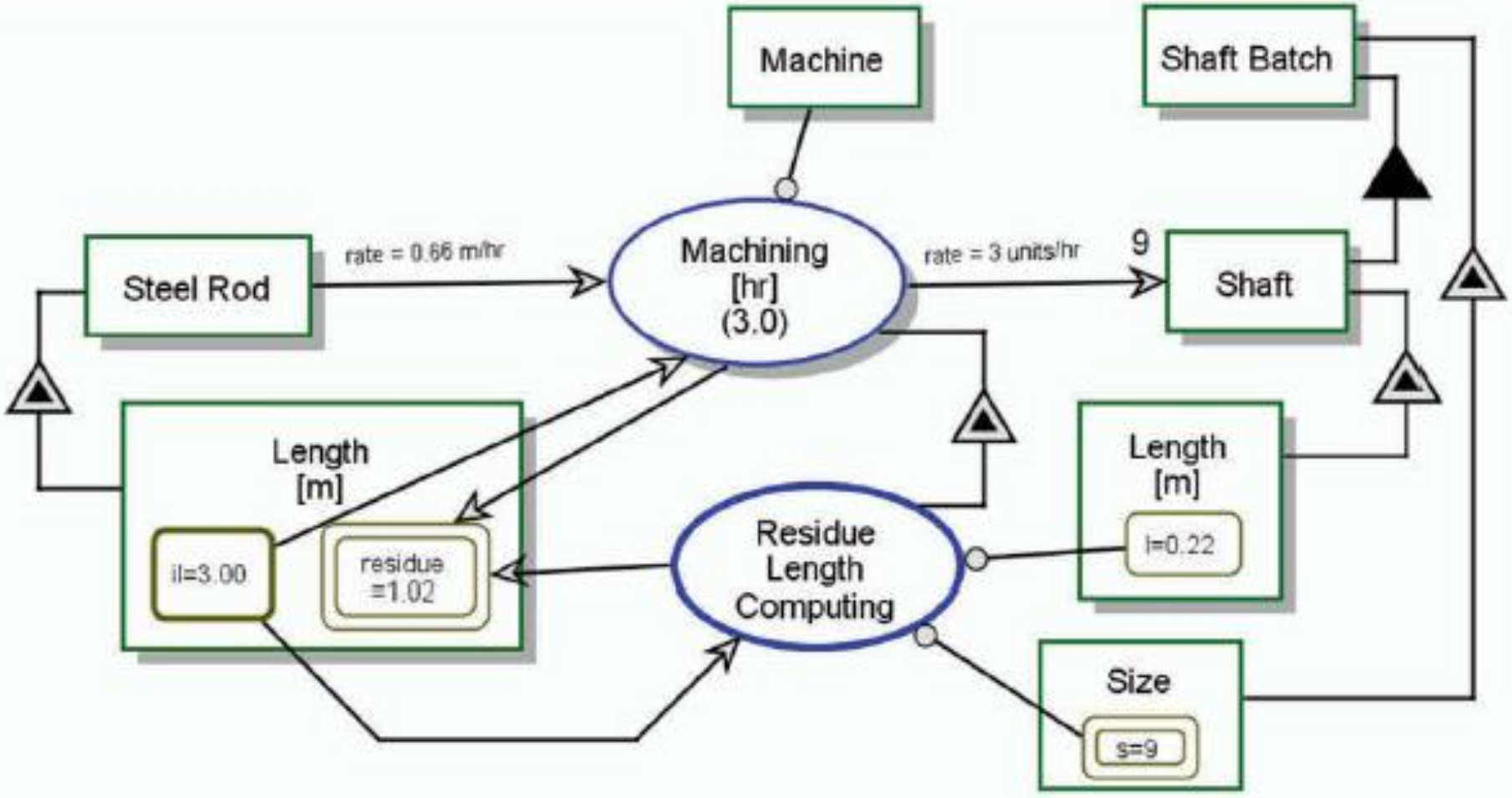
MULTIPLE COPIES



TIMING

- A process may have a **Duration** attribute with a value that expresses units of time. **Duration** may specialize into **Minimal Duration**, **Expected Duration**, and **Maximal Duration.**
  - The **overtime exception link** shall connect the source process with an overtime handling destination process to specify that if at runtime, performance of the source process instance exceeds its **Maximal Duration** value, then an event initiates the destination process.
  - The **undertime exception link** shall connect the source process with an undertime handling destination process to specify that if at runtime, performance of the source process instance takes less than its **Minimal Duration** value, then an event initiates the destination process.

CALCULATIONS



Residue Length Computing requires the value **l=0.22** m of **Length** of **Shaft** and the value of **Size s=9** of **Shaft Batch.**
Residue Length Computing changes the value of **Length** of **Steel Rod** from **il=3.00** m to **residue=1.02.**