Theory of Modelling and Simulation

Ernesto Posse

February 11, 2005

McGill

MSDL

Outline

- ✘ Motivation
- ✘ Hierarchies of System Specification
- ✘ System morphisms
  - ✘ System morphisms at the same level of specification
  - ✘ System morphisms between different levels of specification
- ✘ Analysis and verification
- ✘ Category Theory: the meta-theory

Motivation

- Modelling: description of a (dynamic) system
- Simulation: generation of possible behaviours of a system given a model
- Analysis: reasoning about a system's behaviour

Motivation

- Formalisms for
  - modelling: DEVS, Statecharts, Petri Nets, ODEs, PDEs, etc.
  - analysis: temporal logics, other logics.
  - simulation: state trajectories
- Theory:
  - Foundation
  - Common framework to explain/describe formalisms involved
  - Tools for analysis: general properties of systems and/or formalisms
  - Basis for (software) tools

Progress

✔ Motivation

✘ Hierarchies of System Specification

✘ System morphisms

  - ✘ System morphisms at the same level of specification
  - ✘ System morphisms at different levels of specification

✘ Analysis and verification

✘ Category Theory: the meta-theory

Hierarchies of System Specification

- Model = system specification
- A model can be given at different levels of *abstraction* or *specification*
- Modelling process (by refinement:) from abstract to concrete

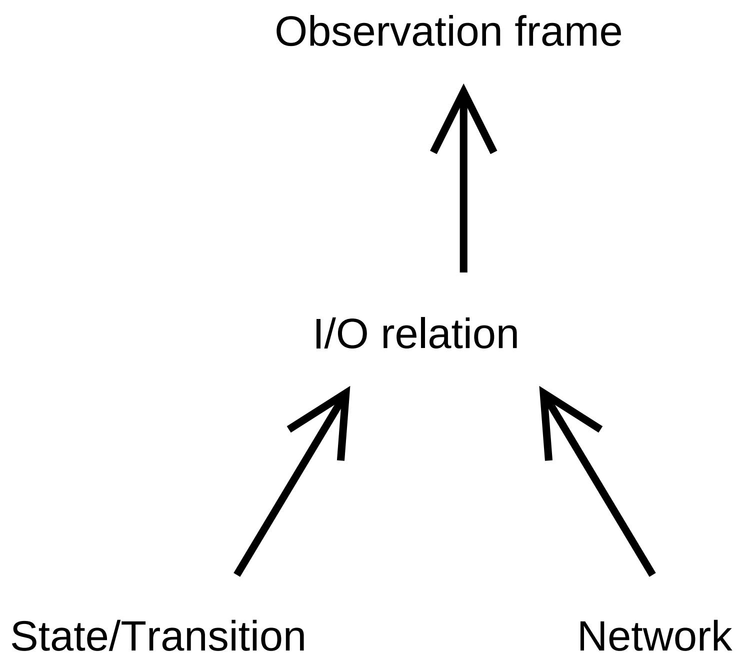
Hierarchies of System Specification

- Dynamic systems: time-varying behaviour
- *Time-base*: ordered set (with a few properties.)
- *Signals* (or *trajectories*): functions from the time-base to some set
- *Segment*: function from a time-interval to some set

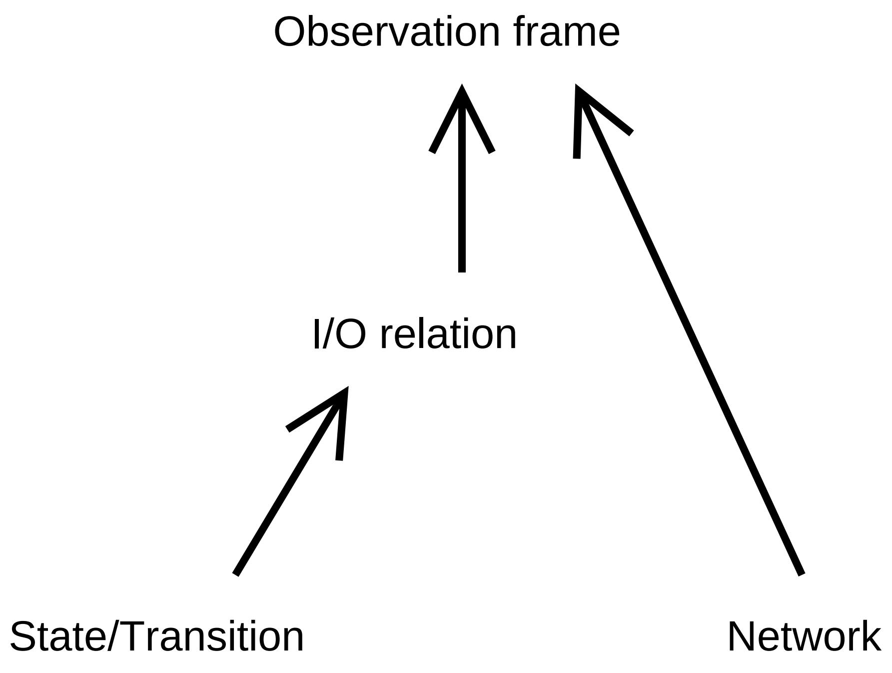
Hierarchies of System Specification

- A simple hierarchy:
  - Observation frame: inputs and outputs
  - I/O relation
  - State/Transition
  - Network

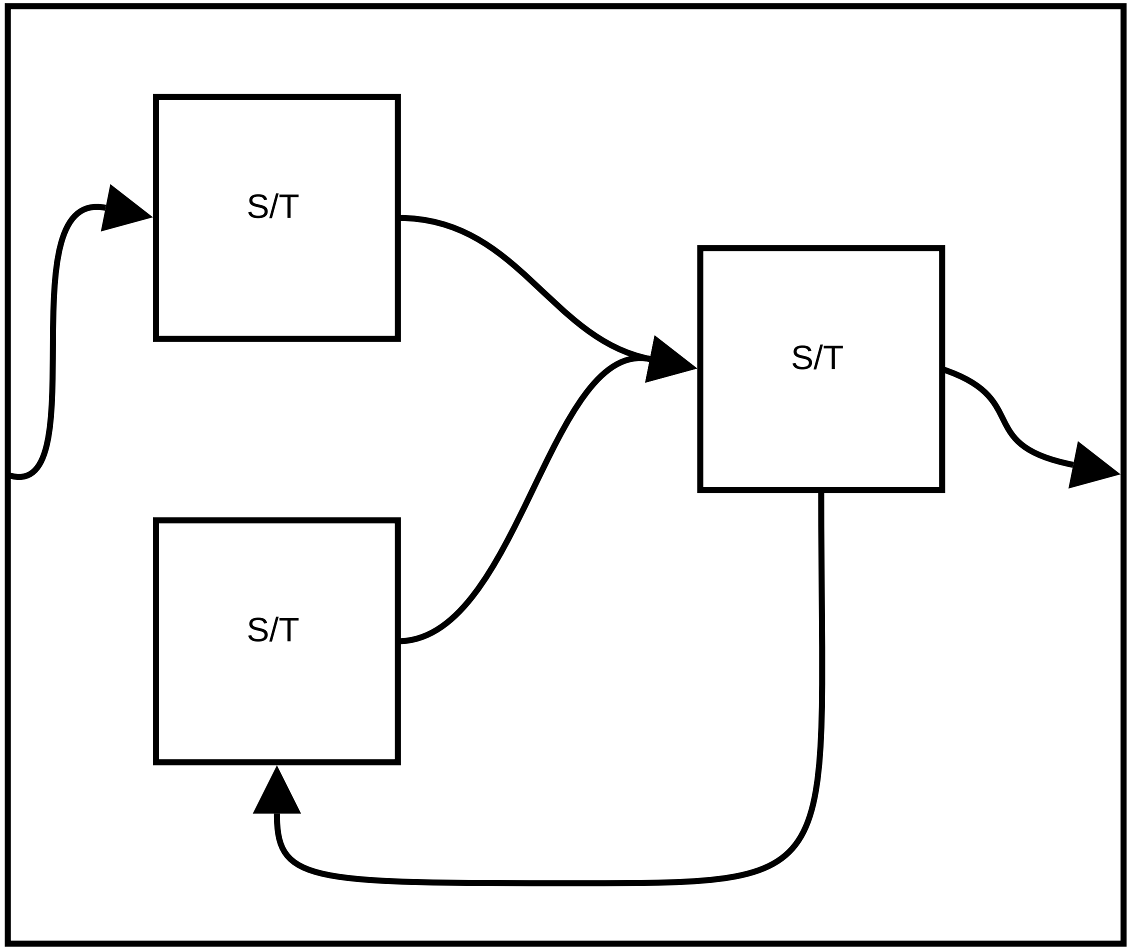
Hierarchies of System Specification



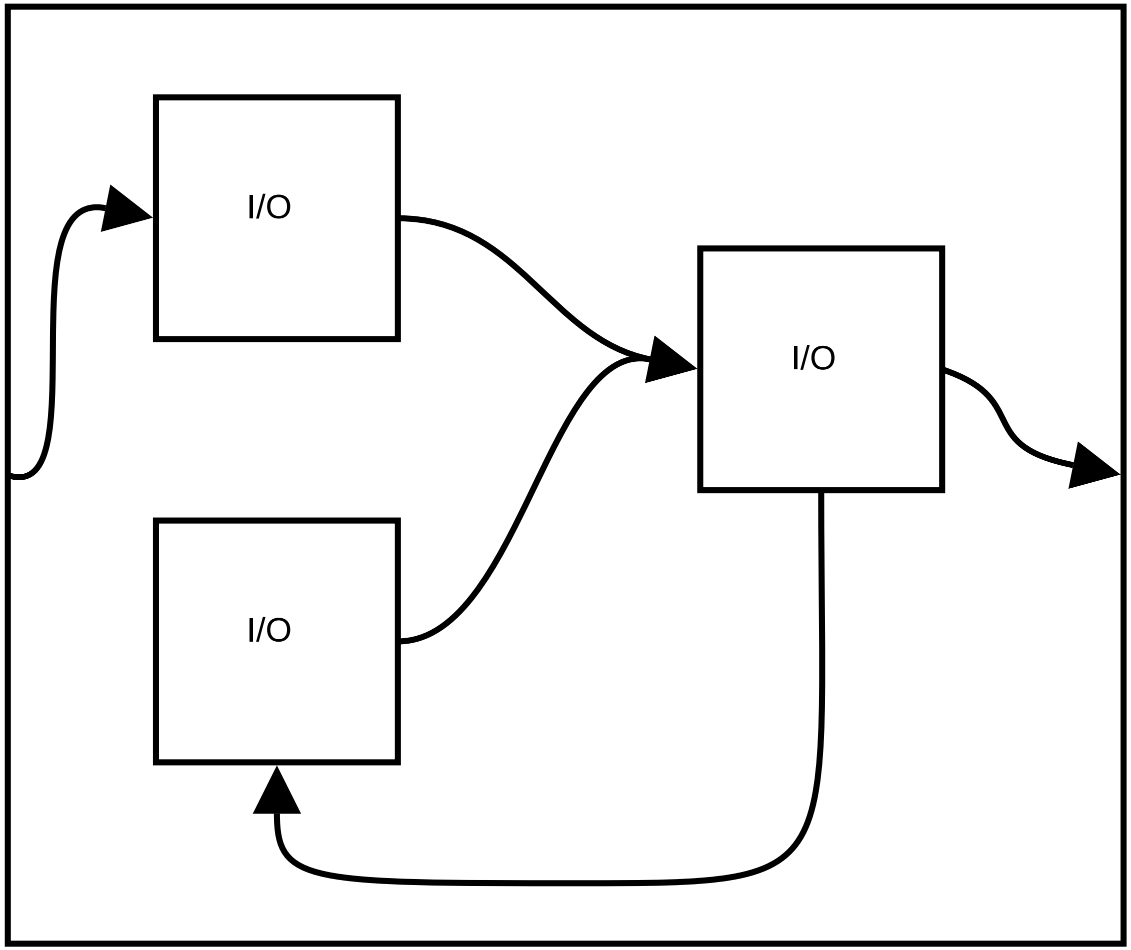
Hierarchies of System Specification



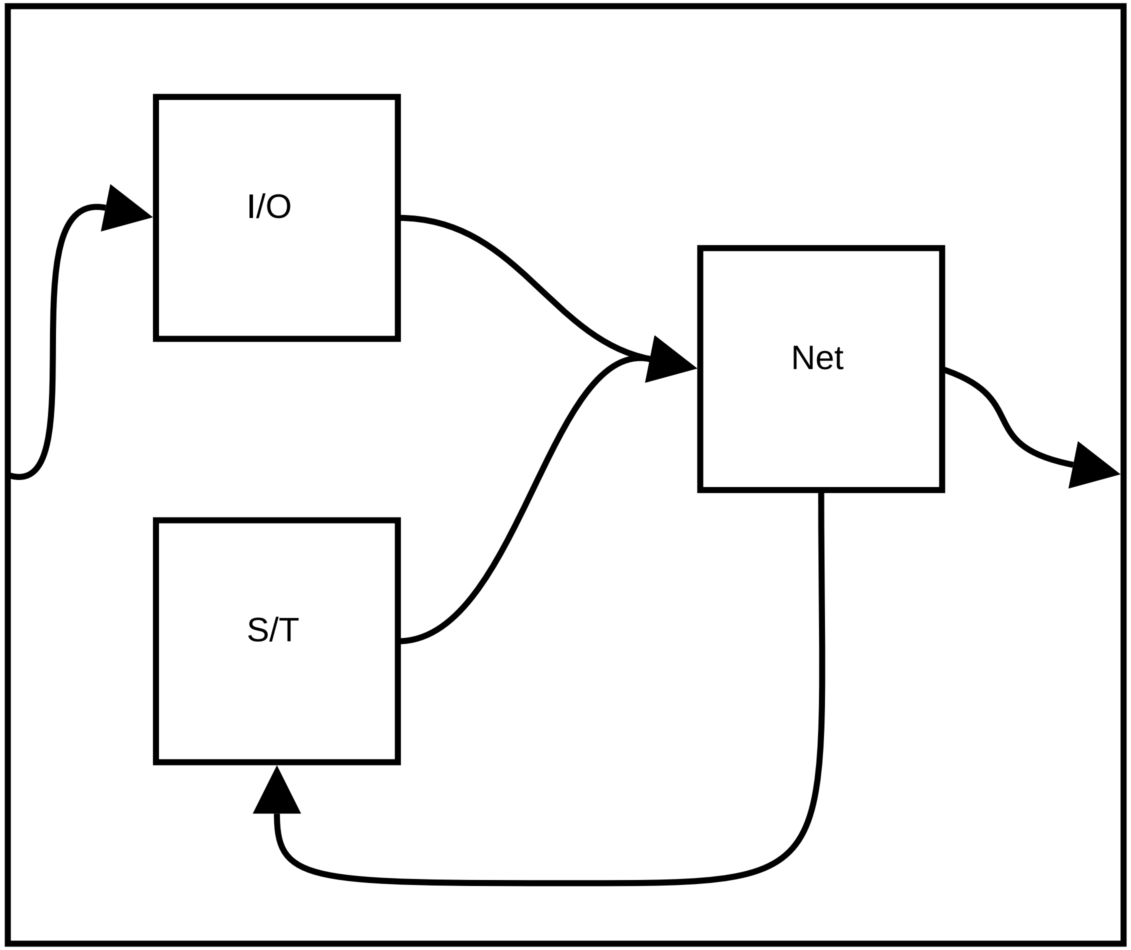
Hierarchies of System Specification



Hierarchies of System Specification



Hierarchies of System Specification



Hierarchies of System Specification

- A general hierarchy of system specification is not a total order but a partial order
- Different kinds of abstraction relationships:
  - Structural
  - Behavioural
- Fitting multiple formalisms in the hierarchy
- Mapping formalisms into the hierarchy
  - Formalisms semantics
  - Multi-formalism modelling

Progress

✔ Motivation

✔ Hierarchies of System Specification

✘ System morphisms

- ✘ System morphisms at the same level of specification
- ✘ System morphisms at different levels of specification

✘ Analysis and verification

✘ Category Theory: the meta-theory

System morphisms

- Some questions:
  - Can I plug-in this component in that network?
  - Can I put system A in place of system B and obtain the same behaviour?
  - I want my system to have this I/O relation. Does this system satisfy it?
  - I know system A has this property. Does system B have that property as well?
- Relate different systems

System morphisms at the same level of specification

- At the Observation frame level:
  - Same time-base
  - Same input and output sets
  - Same interface

System morphisms at the same level of specification

- At the Observation frame level:
  - Equivalent time-base
  - Equivalent input and output sets
  - Equivalent interface

System morphisms at the same level of specification

- At the Observation frame level:
  - Compatible time-base
  - Compatible input and output sets
  - Compatible interface
- Existence of a map between
  - The time-bases (speed)
  - The input and output sets
  - The interfaces

System morphisms at the same level of specification

- At the I/O relation level
  - Same I/O relation
  - Containment
  - Bijection
  - Bijection + transformation

System morphisms at the same level of specification

- At the S/T level
  - What does it mean for a state/transition system to behave in the same way as another?
  - The concepts of simulation and bisimulation

System morphisms at the same level of specification

- A *labelled transition system* (LTS) is a tuple $(S, L, \rightarrow)$ where
  - S is a set of states
  - L is a set of labels (e.g. actions, or conditions)
  - $\rightarrow \subseteq S \times L \times S$ is a transition relation
- We write $p \xrightarrow{a} q$ to mean $(p, a, q) \in \rightarrow$
- An LTS is not a DFA or NFA

System morphisms at the same level of specification

- A *simulation* is a binary relation $R \subseteq S \times S$ such that if $(p, q) \in R$ then whenever

$$p \xrightarrow{a} p'$$

then

$$q \xrightarrow{a} q'$$

and

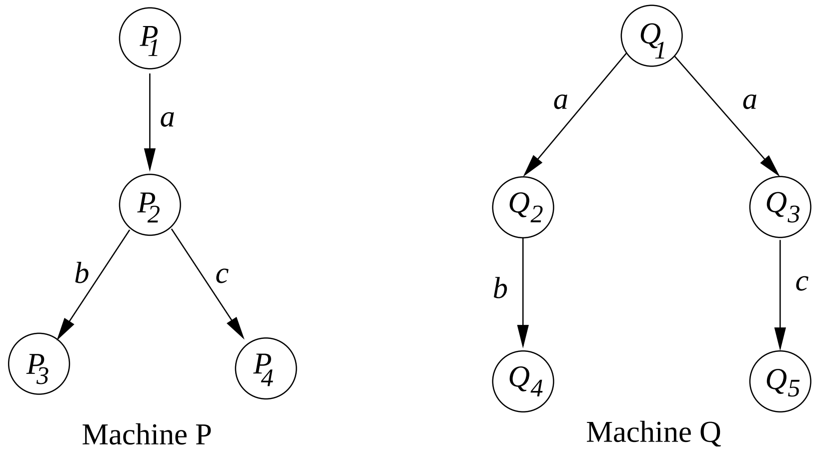
$$(p', q') \in R$$

- p and q are *similar*, written $p \preceq q$ if there is a simulation relation R such that $(p, q) \in R$

System morphisms at the same level of specification

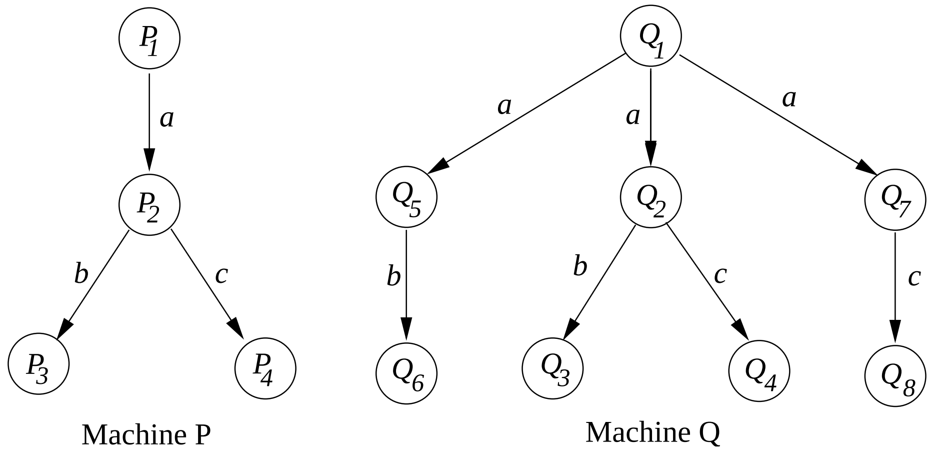
- A *bisimulation* is a binary relation $R \subseteq S \times S$ such that if $(p, q) \in R$ then
  - whenever $p \xrightarrow{a} p'$ then $q \xrightarrow{a} q'$ and $(p', q') \in R$, and
  - whenever $q \xrightarrow{a} q'$ then $p \xrightarrow{a} p'$ and $(p', q') \in R$
- p and q are *bisimilar*, written $p \sim q$ if there is a bisimulation relation R such that $(p, q) \in R$
- Note: If $p \preceq q$ and $q \preceq p$ then it is not necessarily the case that $p \sim q$

System morphisms at the same level of specification



- $Q_1 \preceq P_1$ because there is a simulation R s.t. $(Q_1, P_1) \in R$
- $R = \{(Q_1, P_1), (Q_2, P_2), (Q_3, P_2), (Q_4, P_3), (Q_5, P_4)\}$

System morphisms at the same level of specification

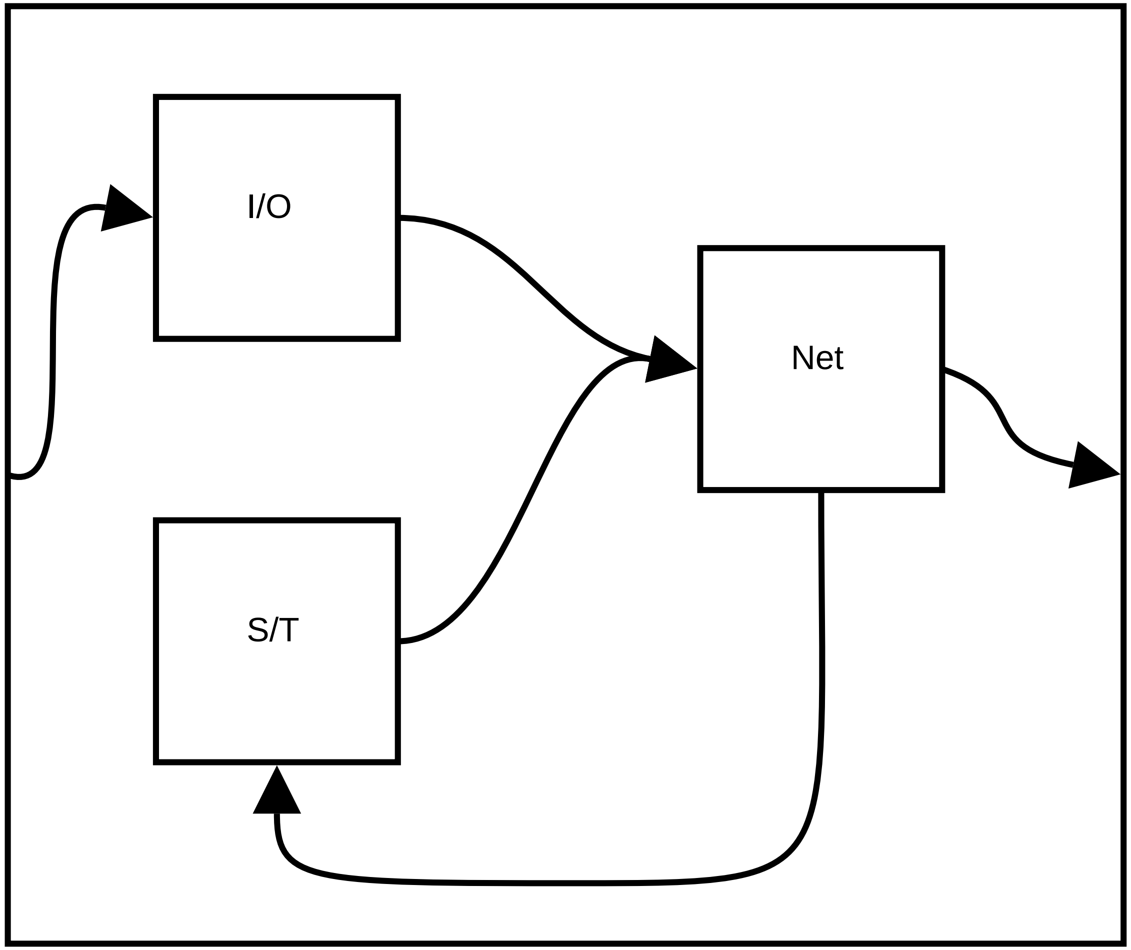


- $Q_1 \preceq P_1$ and $P_1 \preceq Q_1$ but $Q_1 \not\sim P_1$

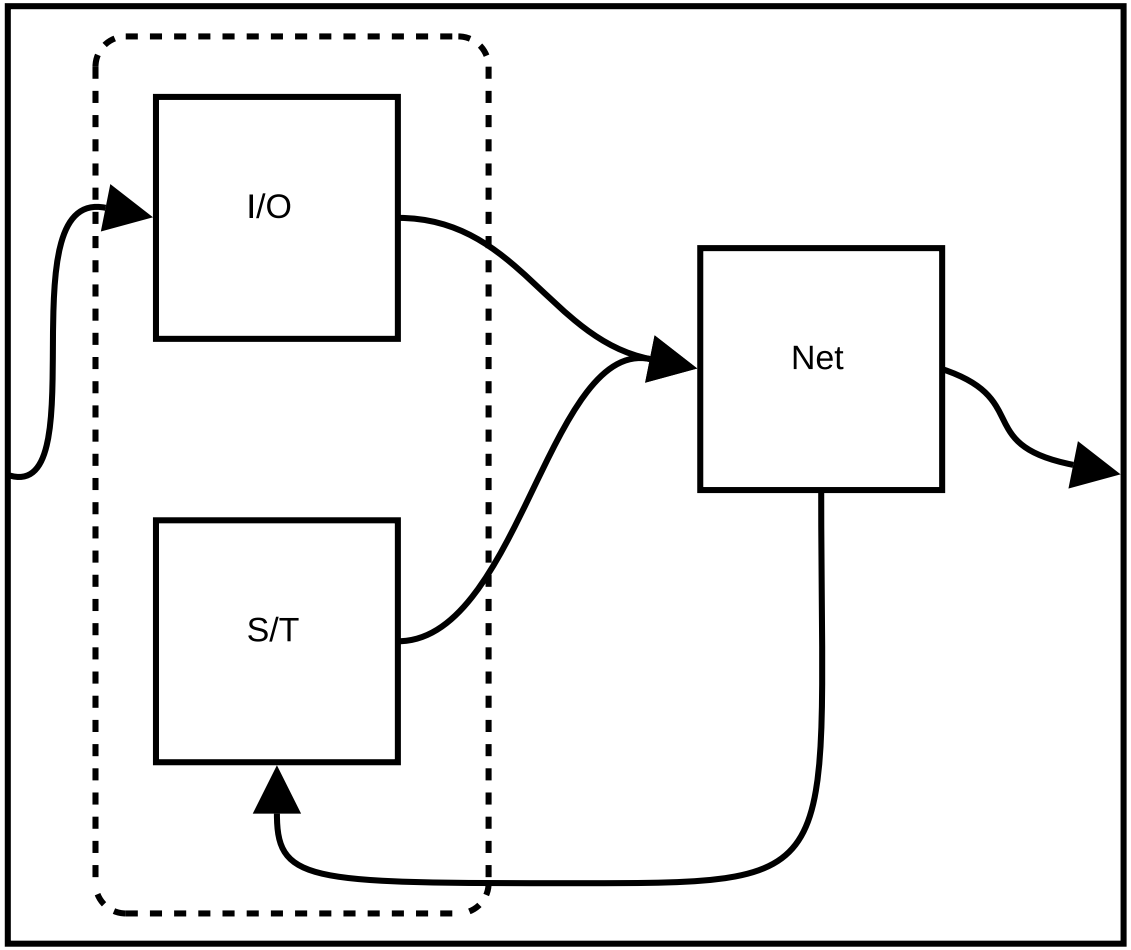
System morphisms at the same level of specification

- At the Network level
  - Matching interfaces
  - Graph-homomorphism

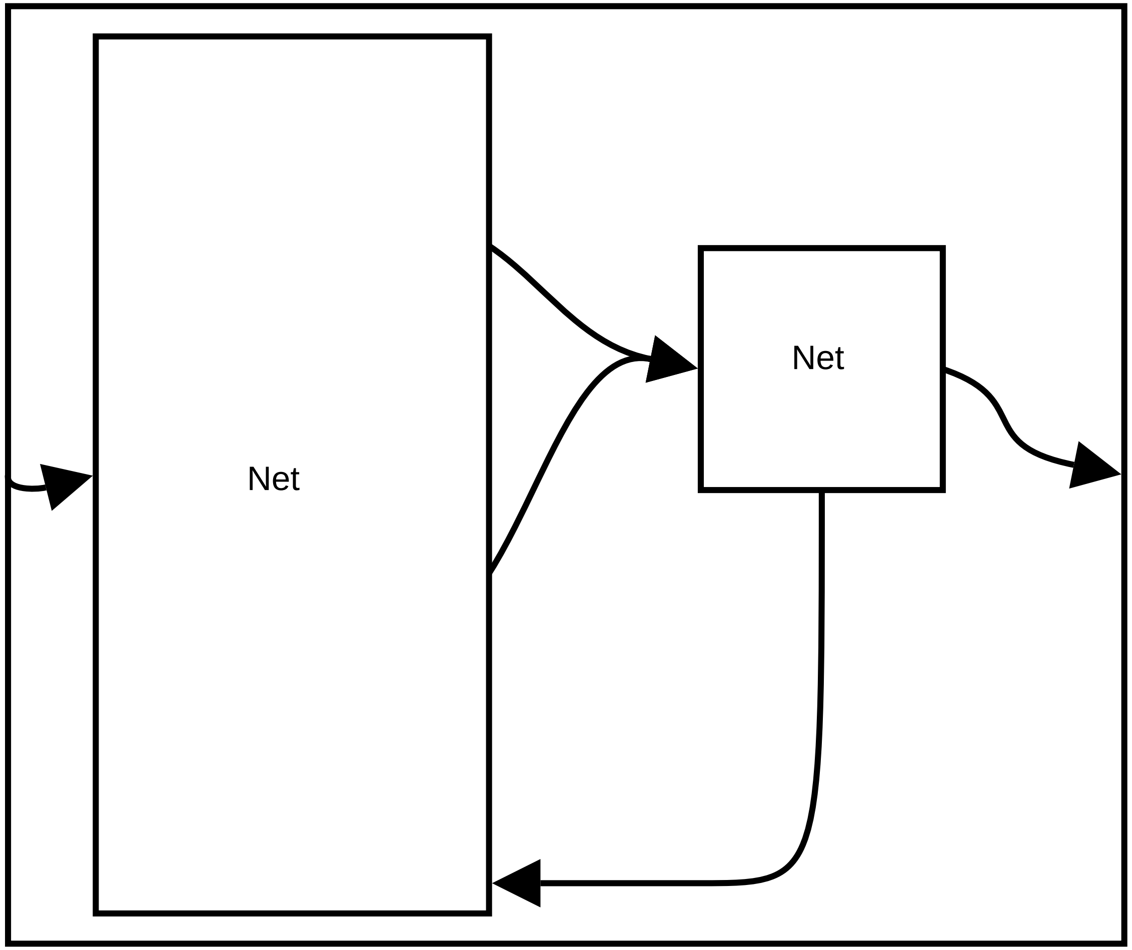
System morphisms at the same level of specification



System morphisms at the same level of specification



System morphisms at the same level of specification



Progress

✔ Motivation

✔ Hierarchies of System Specification

✔ System morphisms

  ✔ System morphisms at the same level of specification

  ✘ System morphisms at different levels of specification

✘ Analysis and verification

✘ Category Theory: the meta-theory

System morphisms at different levels of specification

- Structural
- Behavioural
- Mixed

System morphisms at different levels of specification

- Structural
  - Between Observation Frame and Network
  - Between Networks
- Usually given by an homomorphism
- Answers the question:

  "Can I plug-in this component in that network?"

System morphisms at different levels of specification

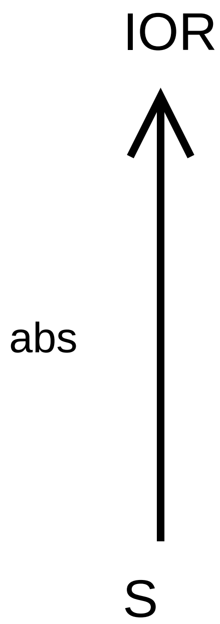
- Behavioural
  - Between I/O relation and State/Transition
- Answers the questions:

  "What is the behaviour of this system?"

  and

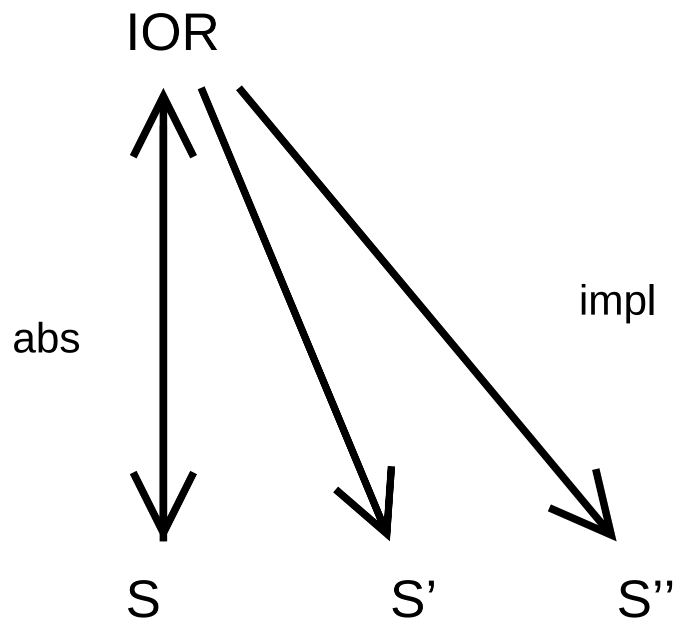
  "Given this I/O relation, does that system satisfy it?"

System morphisms at different levels of specification



- S satisfies IOR

System morphisms at different levels of specification



- IOR is implemented by S, S', S'', etc.

System morphisms at different levels of specification

$$
\begin{tikzcd}
IOR \arrow[r] & IOR' \arrow[d, "impl"] \\
S \arrow[u, "abs"] \arrow[r] & S'
\end{tikzcd}
$$

- $IOR(S) \subseteq IOR(S')$ if and only if $S \preceq S'$
- $IOR(S) = IOR(S')$ if and only if $S \sim S'$

System morphisms at different levels of specification

- Mixed
  - Between I/O relation and Observation Frame
  - Between State/Transition and Observation Frame
  - Between State/Transition and Network

System morphisms at different levels of specification

- Bisimilarity: observational/behavioural equivalence
- If two systems P and Q are equivalent then no observer should be able to distinguish between them
- If a part P of a composite system (network) $C[P]$ is replaced by another equivalent part Q, then the resulting system $C[Q]$ should behave in the same way
- For all contexts $C[-]$, if $P \sim Q$ then $C[P] \sim C[Q]$
- $\sim$ should be a congruence

Progress

✔ Motivation

✔ Hierarchies of System Specification

✔ System morphisms

- ✔ System morphisms at the same level of specification
- ✔ System morphisms at different levels of specification

✘ Analysis and verification

✘ Category Theory: the meta-theory

Analysis and verification

- Analysis: reasoning about systems and formalisms
- Properties:
  - System specific
  - Formalism specific
  - General

Analysis and verification

- Techniques for establishing system specific properties:
  - Manual: by inspection, by formal analysis
  - Automatic: model-checking
- Modal logics: expressing system properties
  - Temporal logics: LTL, CTL, CTL*, etc.
  - Epistemic logics
  - etc.
- Given a system (and a state,) and some formula, determine if it is satisfied or not

Analysis and verification

- Techniques for establishing formalism specific and general properties
  - Manual: by general induction (on the system structure, on the proof of the property, etc.)
  - Automatic: theorem-proving

Progress

- ✔ Motivation
- ✔ Hierarchies of System Specification
- ✔ System morphisms
  - ✔ System morphisms at the same level of specification
  - ✔ System morphisms at different levels of specification
- ✔ Analysis and verification
- ✘ Category Theory: the meta-theory

Category Theory: the meta-theory

- A framework for expressing and relating different mathematical concepts.

- A *category* is a mathematical structure that represents a family of *objects* (mathematical structures) **and** their relationships (morphisms.)

Category Theory: the meta-theory

- Examples:
  - **Set**: the category of sets and functions
  - **Rel**: the category of sets and relations
  - **Mon**: the category of monoids and monoid homomorphisms
  - **Pre**: the category of preorders and monotonic functions
  - **Vec**: vector spaces and linear transformations
  - **Top**: topological spaces and continuous functions
  - **Graph**: the category of graphs and graph-homomorphisms
  - **ST**: state/transition systems and simulations
  - **Prog**: data-types and programs

Category Theory: the meta-theory

- A *functor* is a map between two categories.
- A *natural transformation* is a map between two functors (that go the same way.)
- An *adjunction* is a relation between two functors (that go in opposite ways.)

Category Theory: the meta-theory

- Abstraction and refinement are adjoint functors

Progress

- ✔ Motivation
- ✔ Hierarchies of System Specification
- ✔ System morphisms
  - ✔ System morphisms at the same level of specification
  - ✔ System morphisms at different levels of specification
- ✔ Analysis and verification
- ✔ Category Theory: the meta-theory

The end

TaDa!