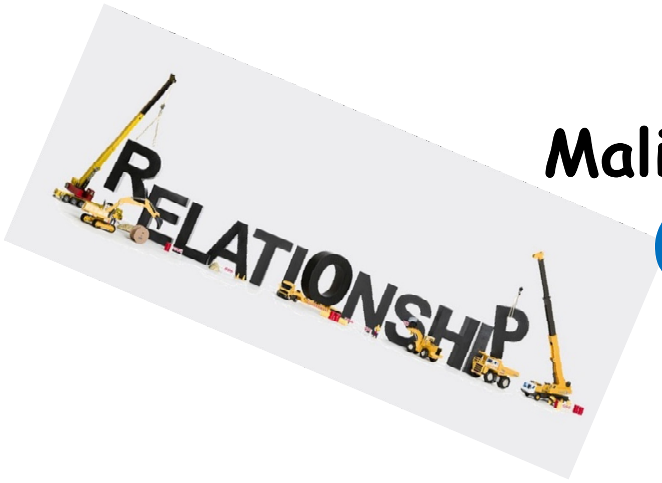
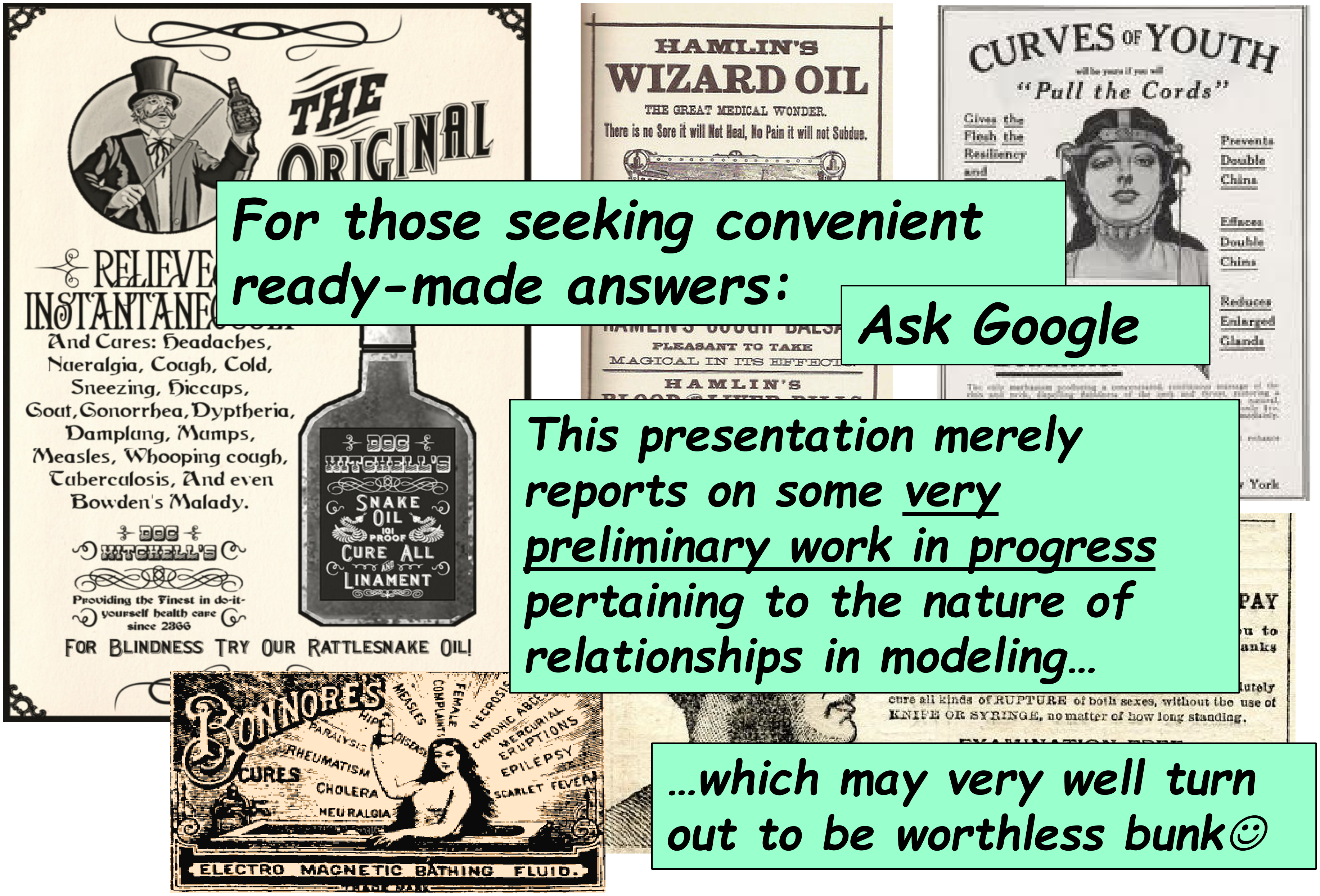
Seeking Meaningful Relationships: A Research Challenge

Bran Selić
Malina Software Corp.
(selic@acm.org)



Caveat Emptor…



THE ORIGINAL
RELIEVES INSTANTANEOUSLY
And Cures: Headaches, Nueralgia, Cough, Cold, Sneezing, Hiccups, Gout, Gonorrhea, Dyptheria, Damplung, Mumps, Measles, Whooping cough, Tuberculosis, And even Bowden's Malady.
DOC MITCHELL'S
Providing the Finest in do-it-yourself health care since 2366
DOC MITCHELL'S SNAKE OIL 101 PROOF CURE ALL AND LINAMENT
FOR BLINDNESS TRY OUR RATTLESNAKE OIL!
HAMLIN'S WIZARD OIL
THE GREAT MEDICAL WONDER.
There is no Sore it will Not Heal, No Pain it will not Subdue.
PLEASANT TO TAKE
MAGICAL IN ITS EFFECT
HAMLIN'S
CURVES OF YOUTH
will be yours if you will
"Pull the Cords"
Gives the Flesh the Resiliency and
Prevents Double Chins
Effaces Double Chins
Reduces Enlarged Glands
PAY
cure all kinds of RUPTURE of both sexes, without the use of KNIFE OR SYRINGE, no matter of how long standing.
BONNORE'S CURES
MEASLES
HIP DISEASE
PARALYSIS
RHEUMATISM
CHOLERA
NEURALGIA
FEMALE COMPLAINTS
NECROSIS
CHRONIC ABCESSES
MERCURIAL ERUPTIONS
EPILEPSY
SCARLET FEVER
ELECTRO MAGNETIC BATHING FLUID.

For those seeking convenient ready-made answers:

Ask Google

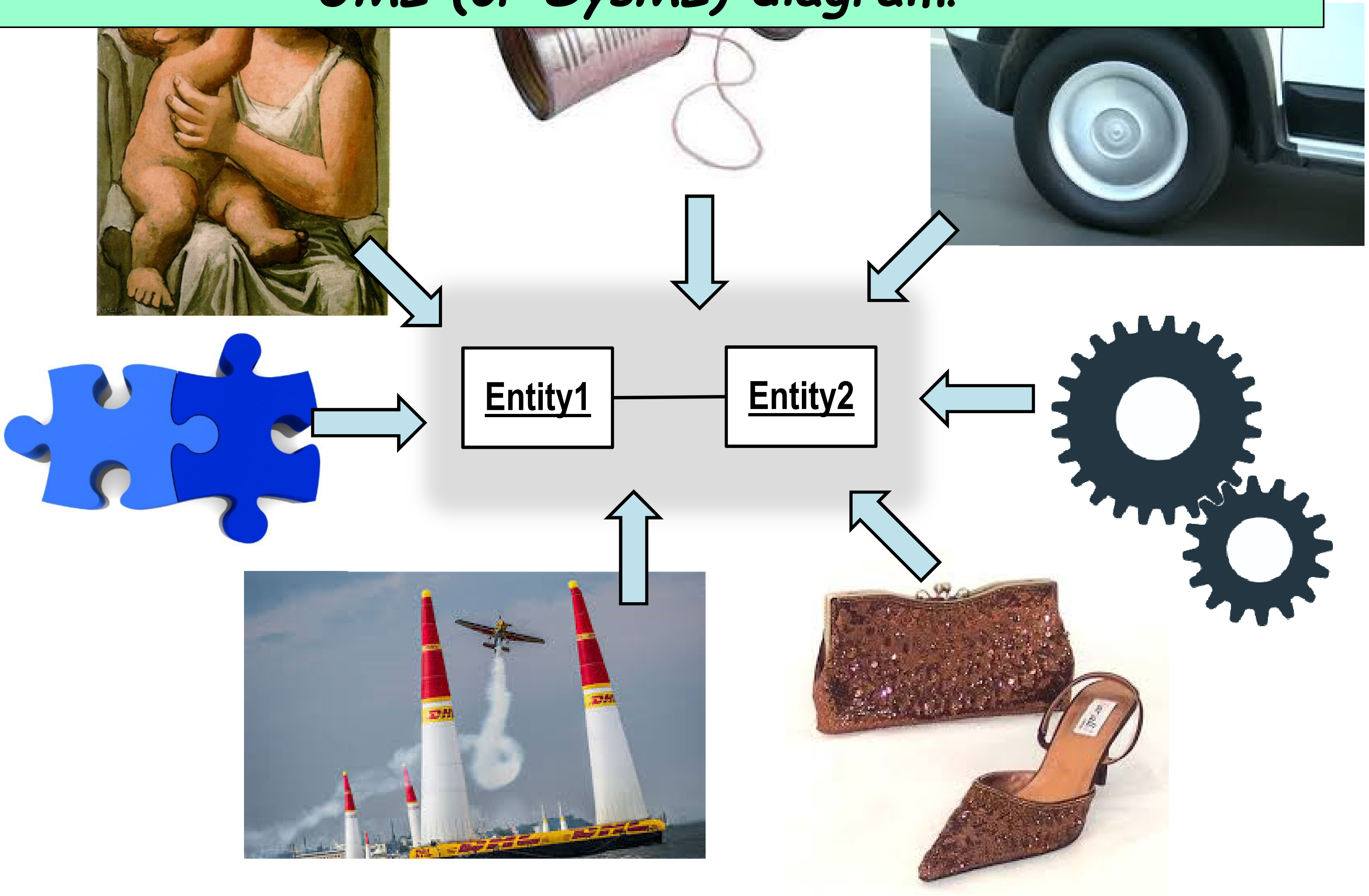
This presentation merely reports on some ***very preliminary work in progress*** *pertaining to the nature of relationships in modeling…*

…which may very well turn out to be worthless bunk☺

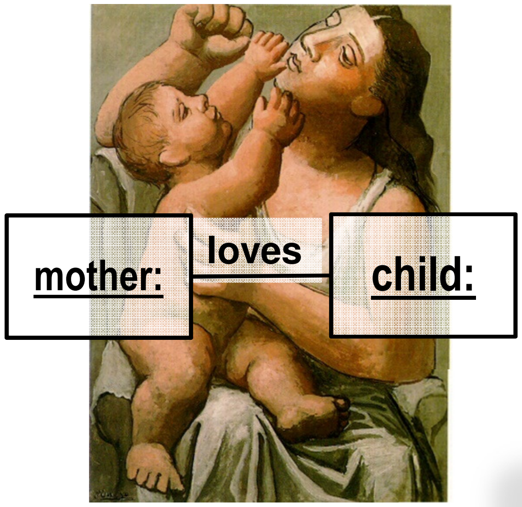
My Concern

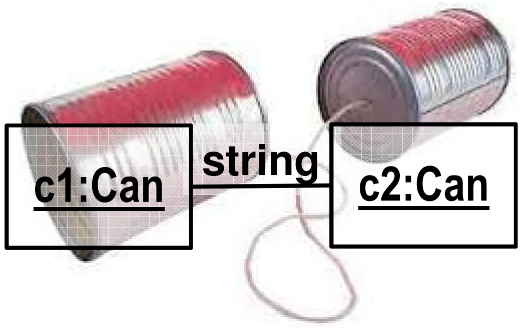
Q: What is Common to All of These?

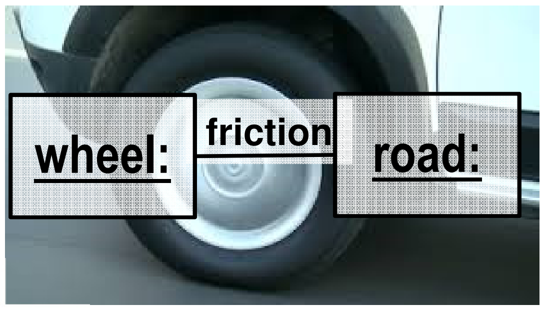
A: They can all can be represented by the same UML (or SysML) diagram!

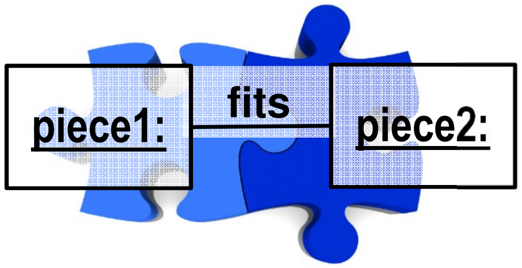


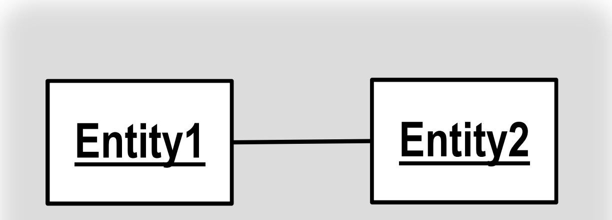
An Abstraction Taken Too Far?

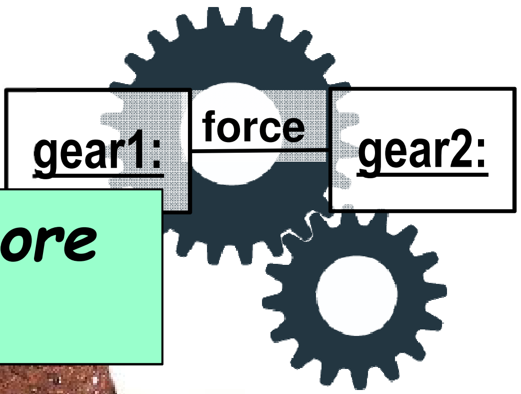




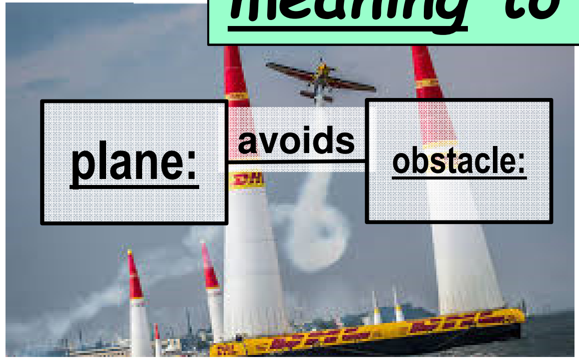


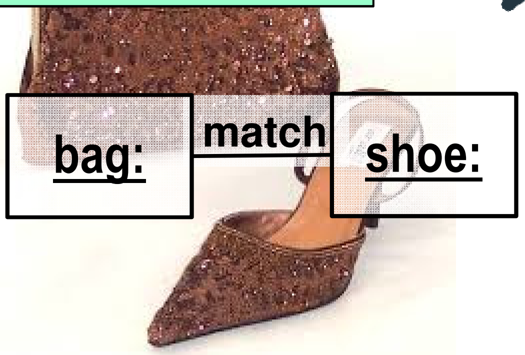






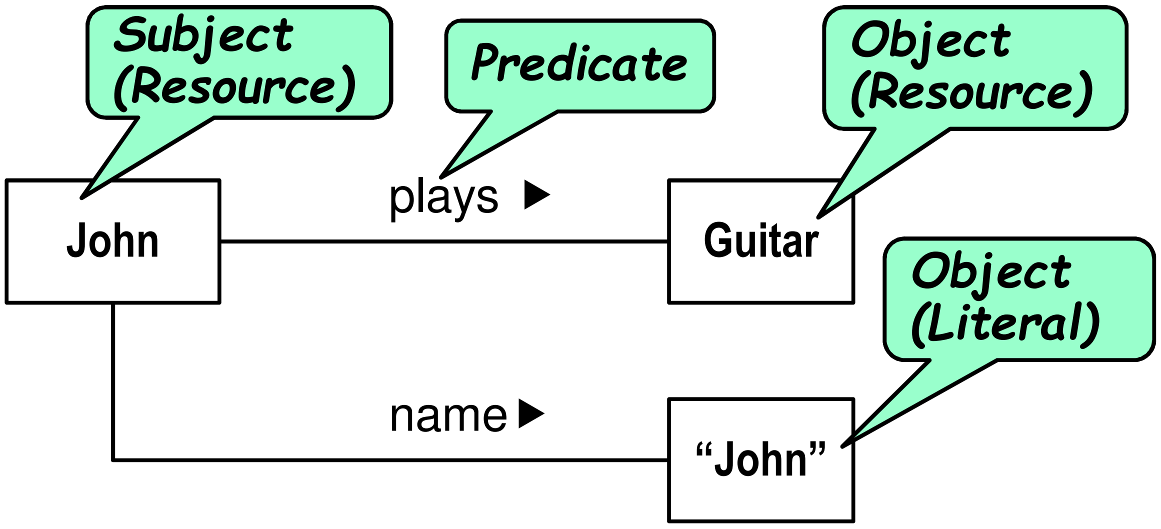
How can we attach more ***meaning*** *to the lines?*





What About the "Semantic Web"?

- RDF, RDFS, OWL 2 [www.w3.org]
  - Predefine a set of general relationship types ("predicates") used in defining ontologies; e.g.:
    - owl:sameAs, owl:differentFrom, rdfs:subClassOf, rdfs:label, owl:hasValue, etc.



- Semantic interpretation of such schemas is external: left up to the code that traverses them

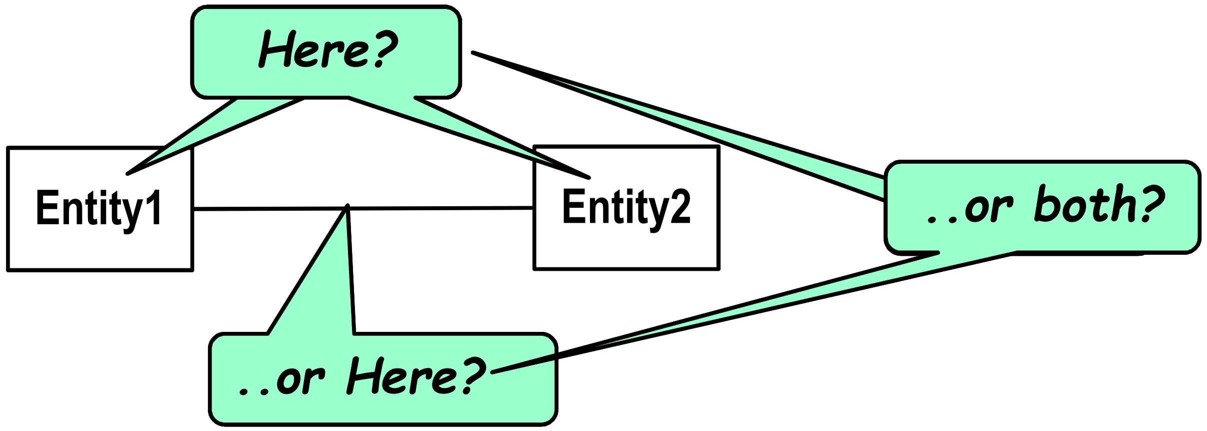
On Meaning…

- **From the "RDF Semantics: document [http://www.w3.org/TR/2004/REC-rdf-mt-20040210/]**

"Exactly what is considered to be the 'meaning' of an assertion in RDF or RDFS in some broad sense may depend on many factors, including social conventions, comments in natural language or links to other content-bearing documents. Much of this meaning will be inaccessible to machine processing and is mentioned here only to emphasize that the formal semantics described in this document is not intended to provide a full analysis of 'meaning' in this broad sense…"

Q: Is it meaningful to talk about "meaning" (semantics) without being explicit about behavior?

Where Do Relationships Behave?



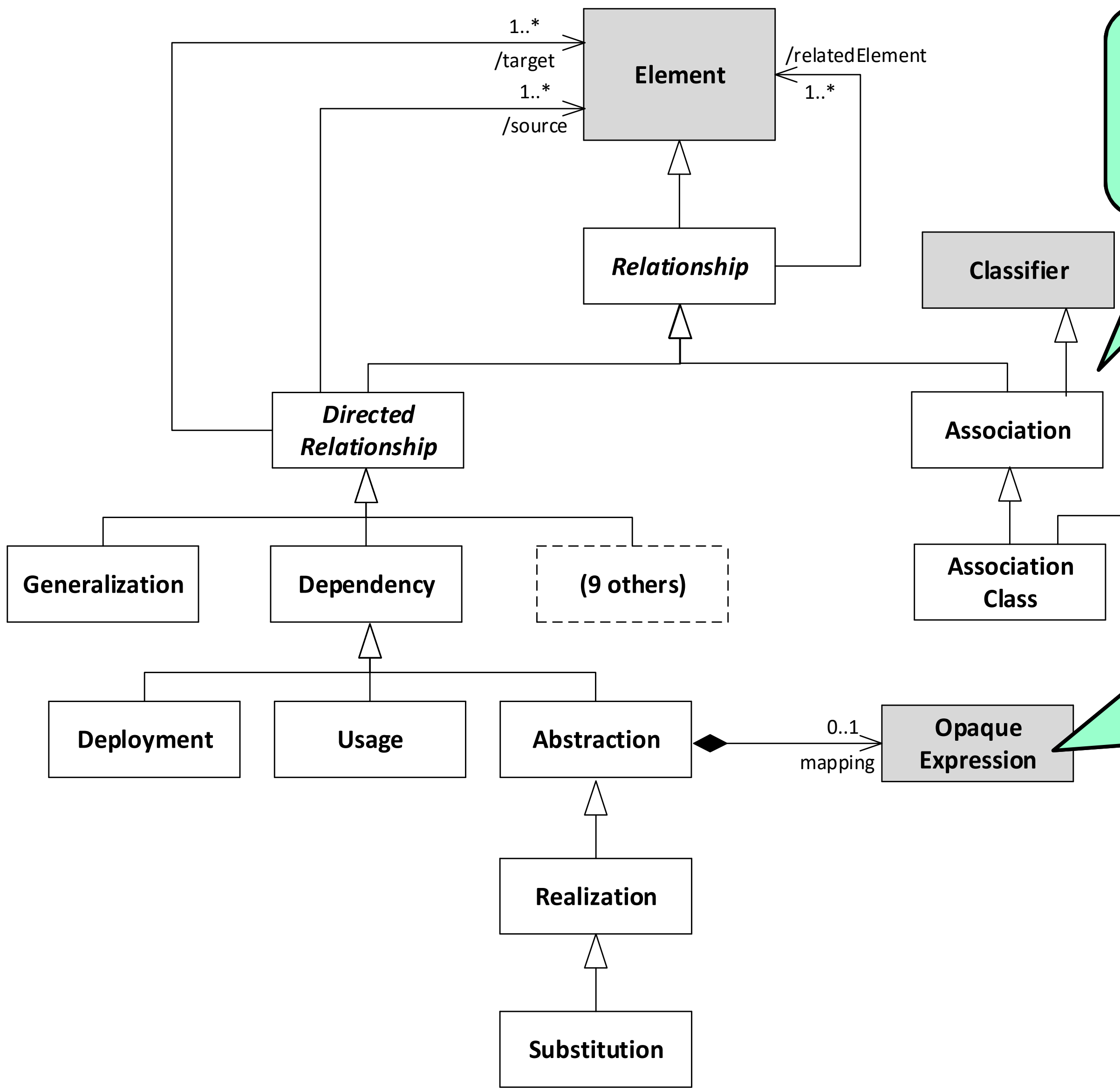
Object behavior vs. Behavior that an object participates in

The Research Challenges

- **How can we provide more information about the meaning of relationships in models?**
  - How much of this can be within (self-contained) the model itself?
  - How should the meaning of relationships be specified?
  - What should be the core set of pre-defined primitive relationships ("axiom relationships")?
- **Rationale:**
  - Computational (automation) viewpoint: more effective processing of information
  - Enterprise (human reader) viewpoint: better understanding of models

Prior Work

The UML Model of Relationships

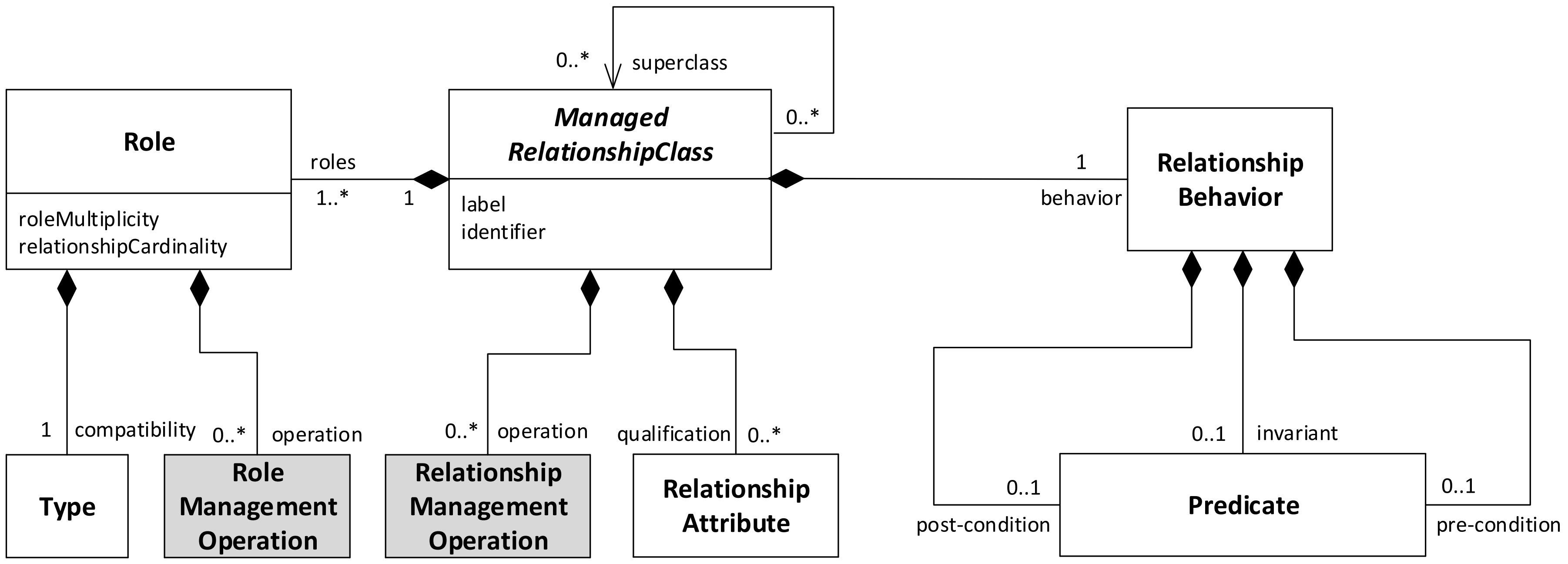


NB: Only Associations are classifiers!

NB: Only Abstractions have provision for additional information

ISO General (Managed) Relationship Model

- ISO General Relationship Model (ISO/IEC 10165-7)
  - Also as: ITU-T Recommendation X.725 (11/95) [free!]



...a contract-based approach

RM-ODP on Contracts

- ISO Standard Reference Model of Open Distributed Processing [ISO-IEC-10746-x]
- Contract: An agreement governing part of the collective behaviour of a set of objects. A contract specifies obligations, permissions and prohibitions for the objects involved. The specification of a contract may include:
  a) a specification of the different roles that objects involved in the contract may assume, and the interfaces associated with the roles;
  b) quality of service attributes;
  c) indications of duration or periods of validity;
  d) indications of behaviour which invalidates the contract;
  e) liveness and safety conditions.

Example: GRM Composition Relationship

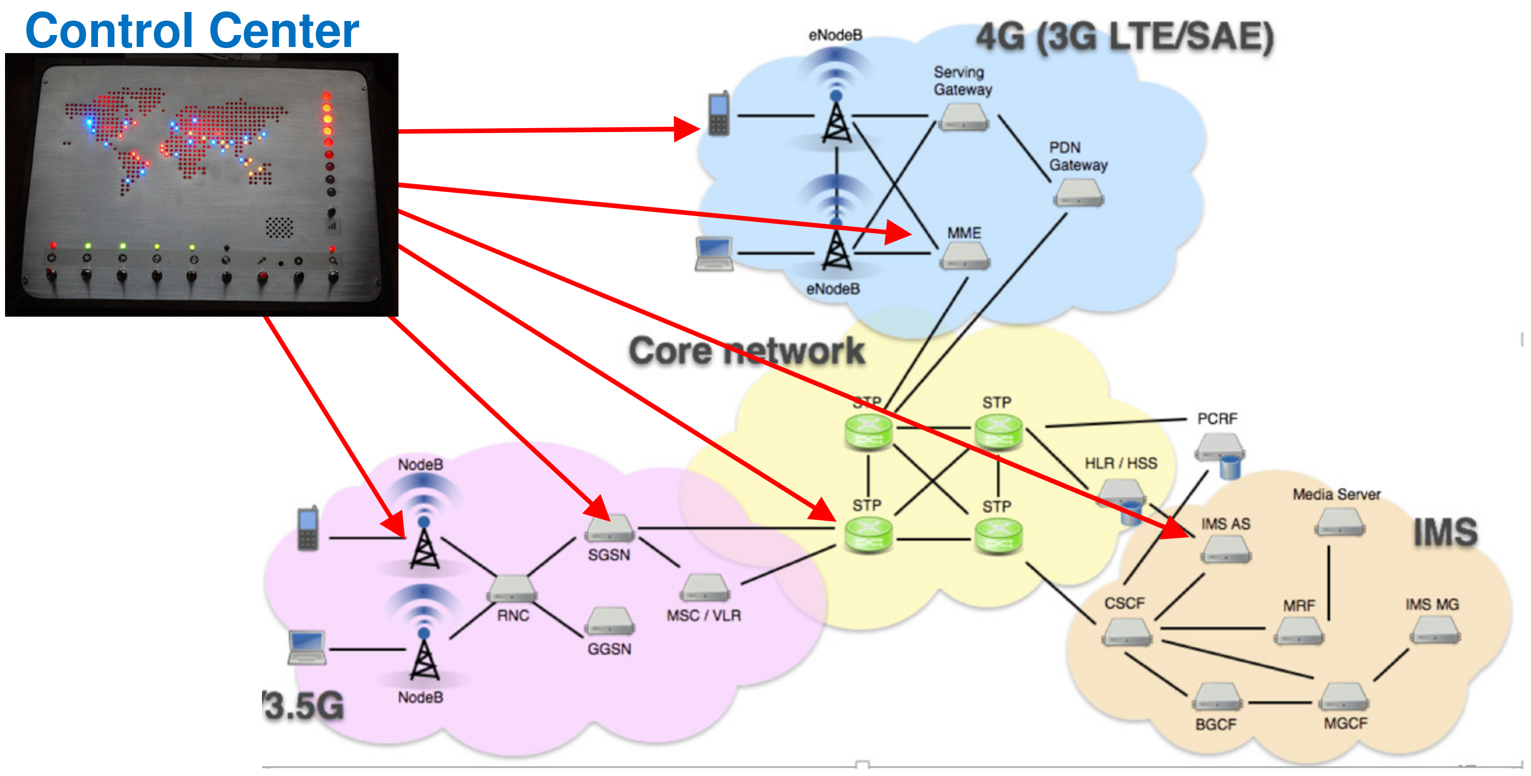
- **"General composition relationship behavior (invariant):**

"The existence of an instance of this relationship class implies the existence of exactly one participant in the composite role and one or more participants in the component role. At least one property of the composite participant is such that it depends upon properties of the components. At least the identity of the composite participant is such that it is independent of the existence or properties of the components; that is, creating, updating, or deleting any component does not change the identity of the composite."

-- pg.37, **ITU-T Recommendation X.725 (11/95)**

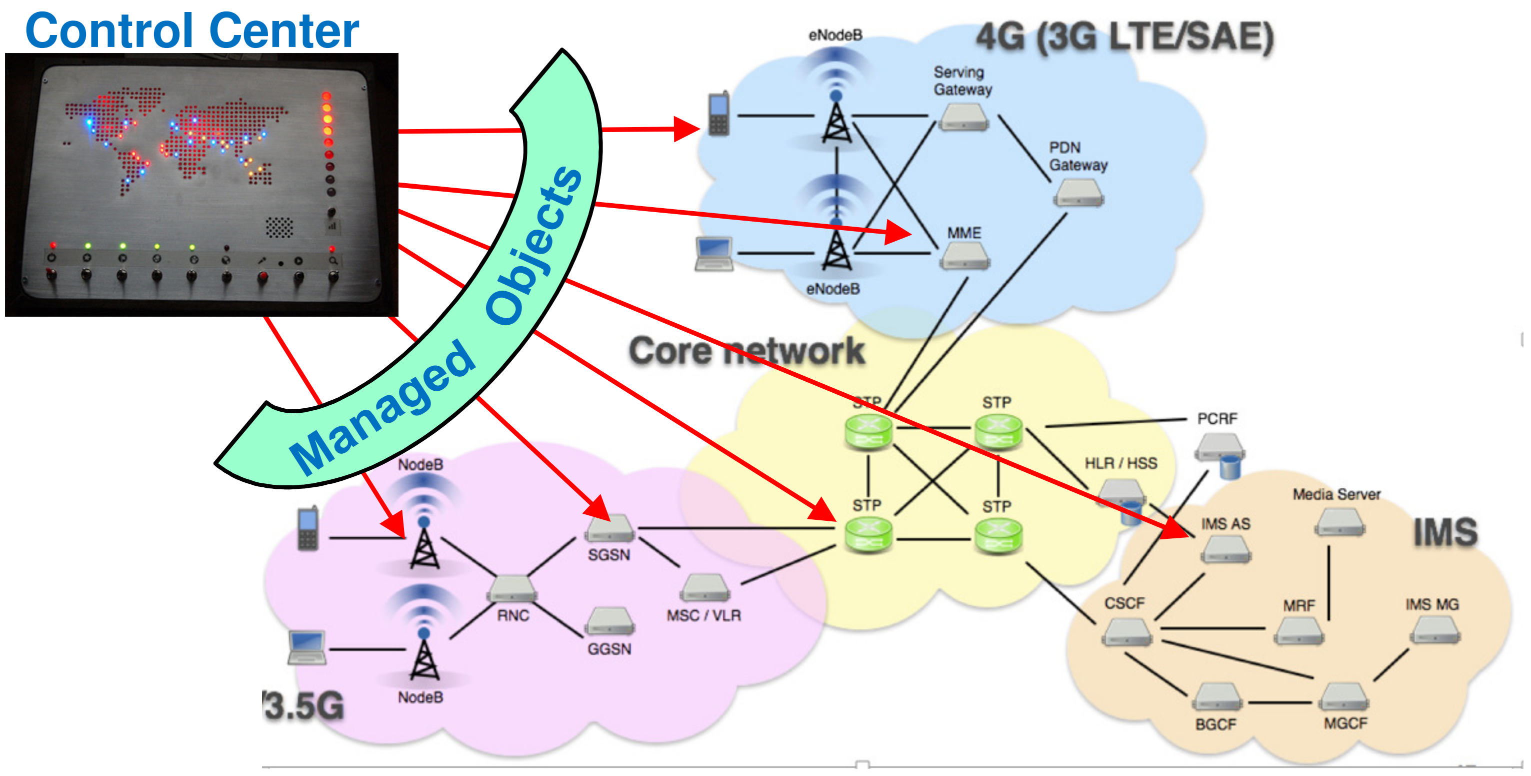
Open Systems Management (Control)

- Open systems are heterogeneous by definition:
  - Different operating principles, architectures, implementation technologies, etc.
- *Q: How do we control such highly complex and diverse systems using a __common control policy__?*



Managed Object Concept

- **A virtualization layer that provides a homogeneous view of a heterogeneous network**
  - All controlled (managed) entities mapped to a common "<u>managed object</u>" model

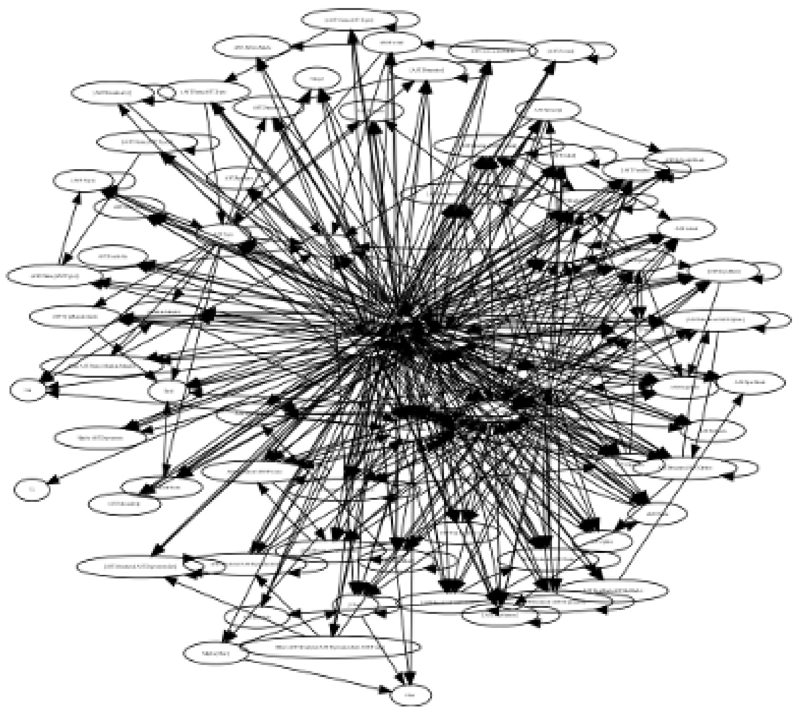


Management Operations for Relationships

- ISO/IEC 10165-7 General Relationship Model (1995)
  - Also as: ITU-T Specification X.725 [free!]
- ESTABLISH (<participantId:class>*)
  - Creates a new relationship with stated participants in appropriate roles
- TERMINATE (<relationshipId>)
  - Destroys the relationship
- BIND (<participantId:class, role>*)
  - Adds a participants in the specified role to a relationship
- UNBIND (<participantId:class, role>*)
  - Release participant in the specified role from the relationship
- QUERY (<operation [,<role>]>)
  - <operation>-specific query of relationship or role(s)
- NOTIFY
  - Report abut events related to the relationship

Do We Really Need to Manage Relationships?

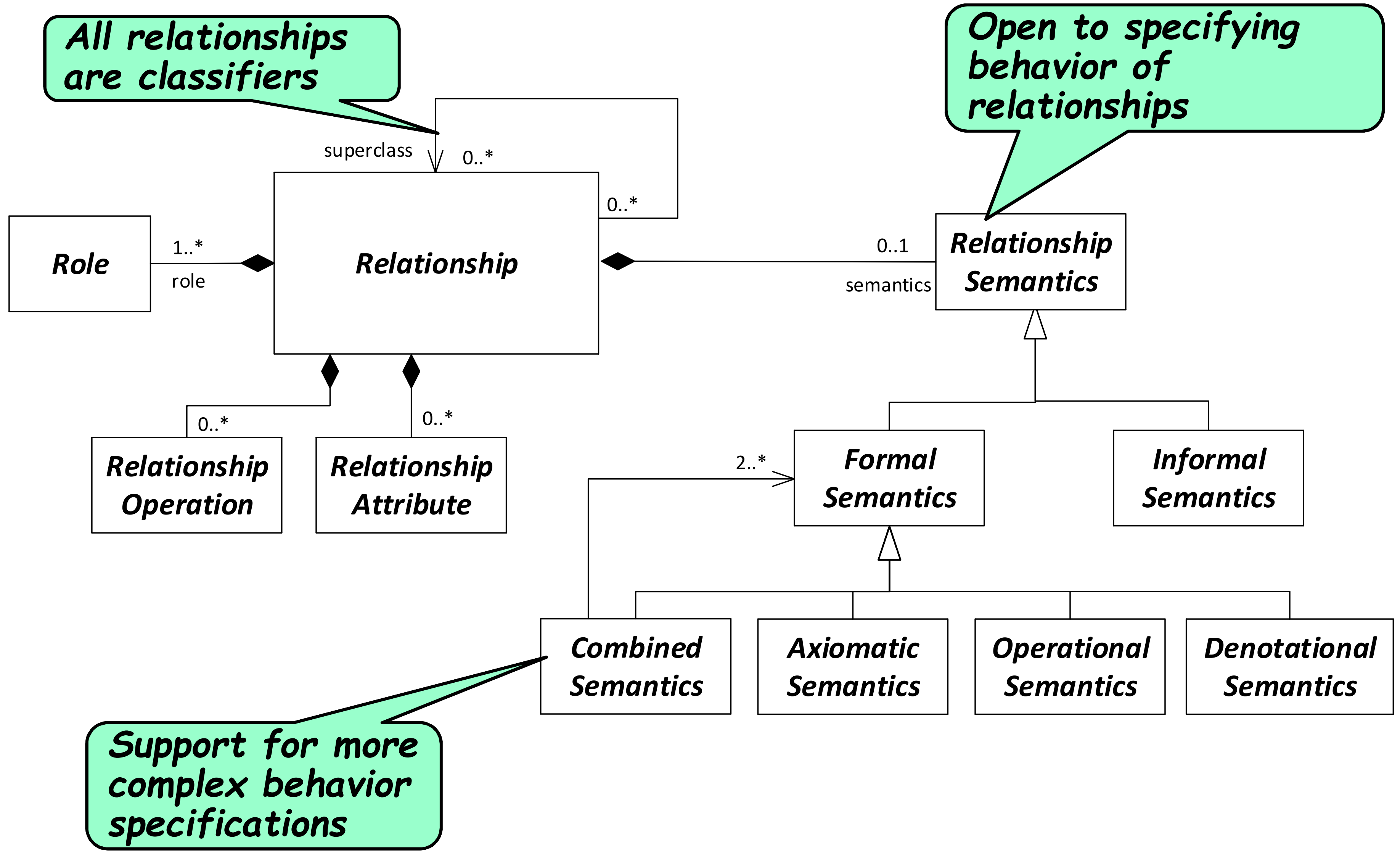
It might prove useful…



- The degree of interconnection in even moderately complex real-world system models can be overwhelming
  - E.g.: traceability links
  - Ultimately, everything is connected to everything else
- Basis for a systematic way of coping with change

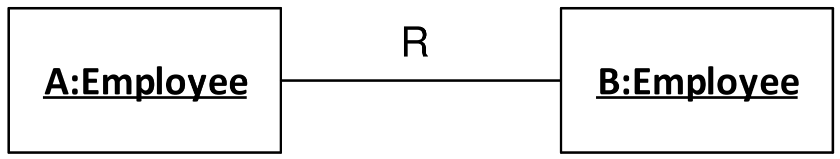
A Preliminary Proposal

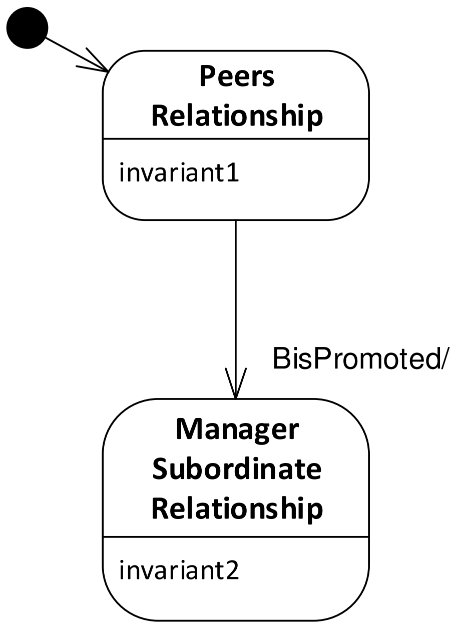
Proposal: A General Relationship Model



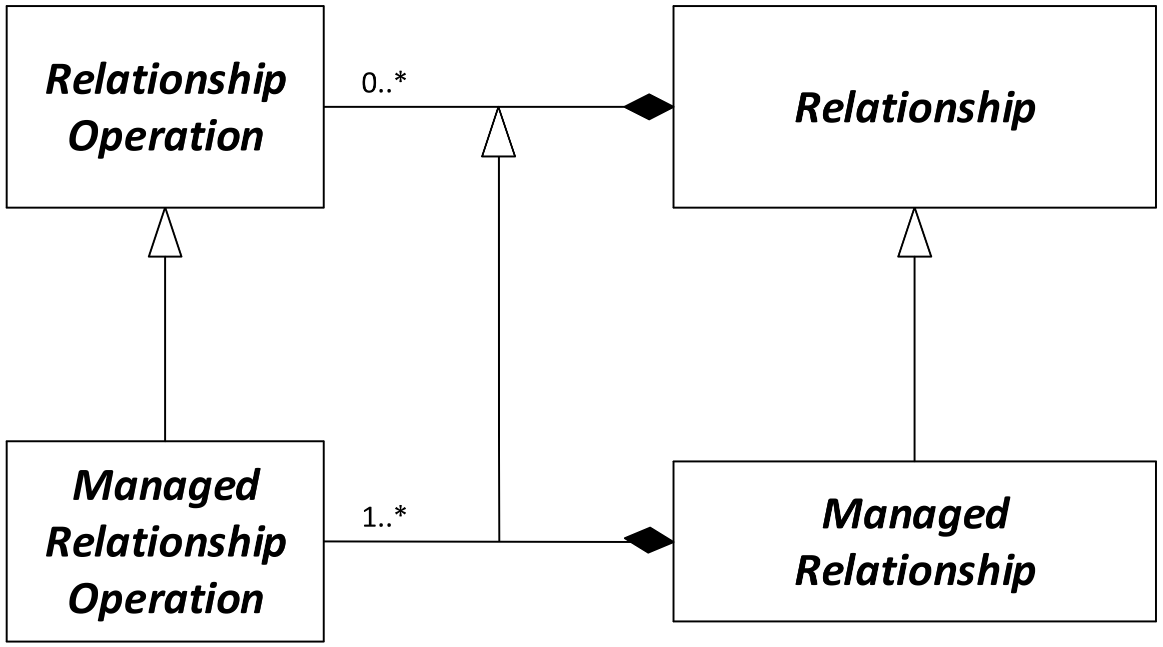
Modeling Dynamic Relationships

- Epoch [RM-ODP]: a span of time during which an object behaves in a certain manner
- Use a combination of different specifications
  - E.g., using state machines in combination with other types of specifications (invariants, etc.)
  - Example, reclassifying a relationship instance:

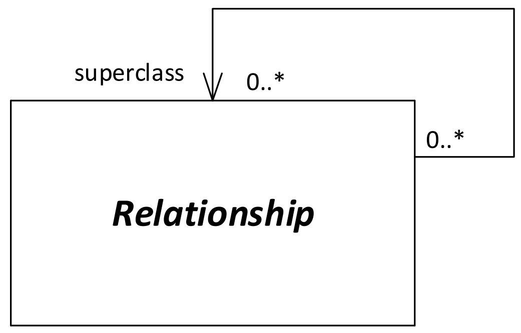




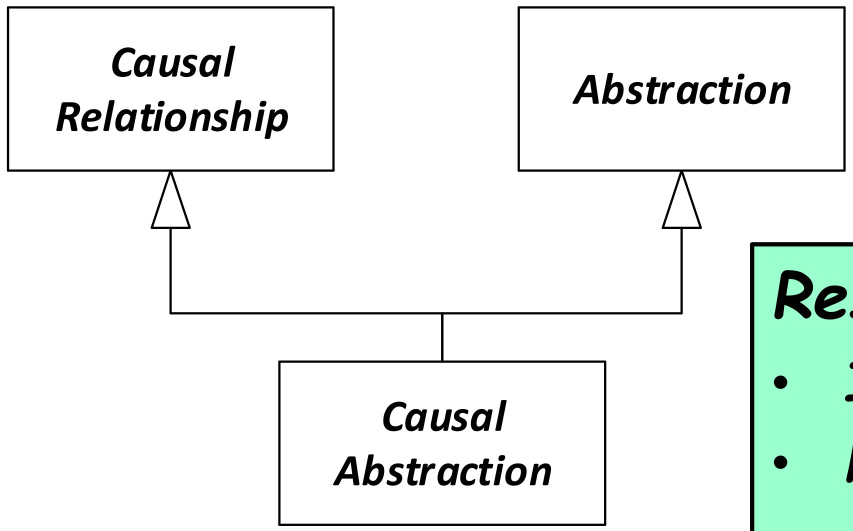
Managed Relationships



Relationships with Complex Semantics



- By combining more primitive relationships via inheritance?

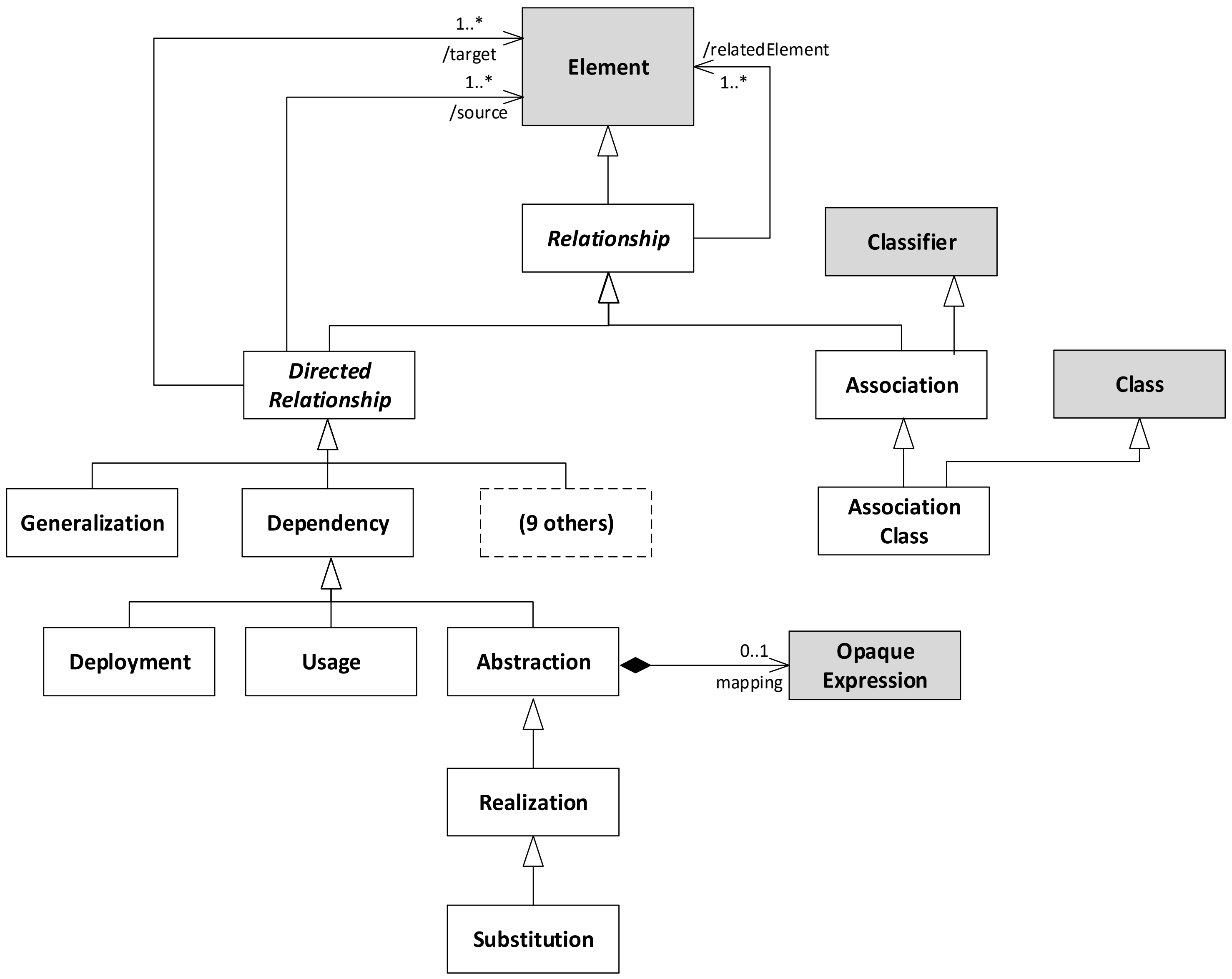


Research questions:

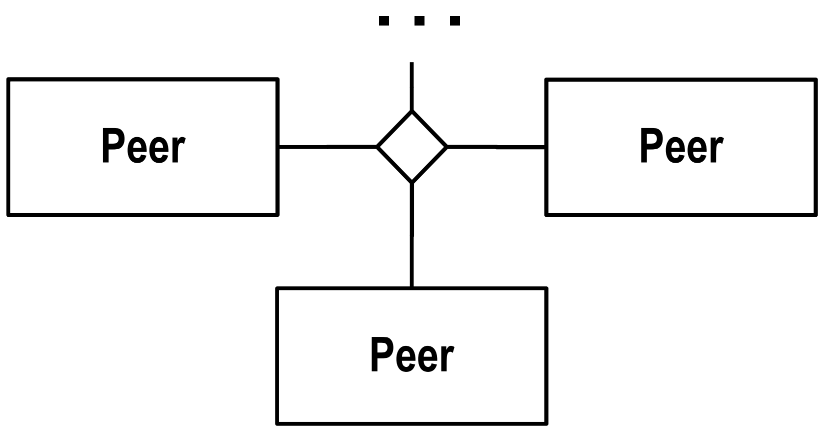
- *Is it practical?*
- *How do we combine roles?*
- *etc.*

What should be the core set of "axiom relationships"?

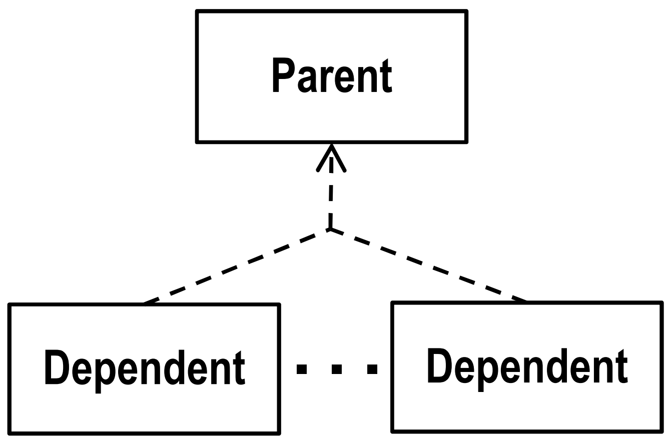
The UML Taxonomy



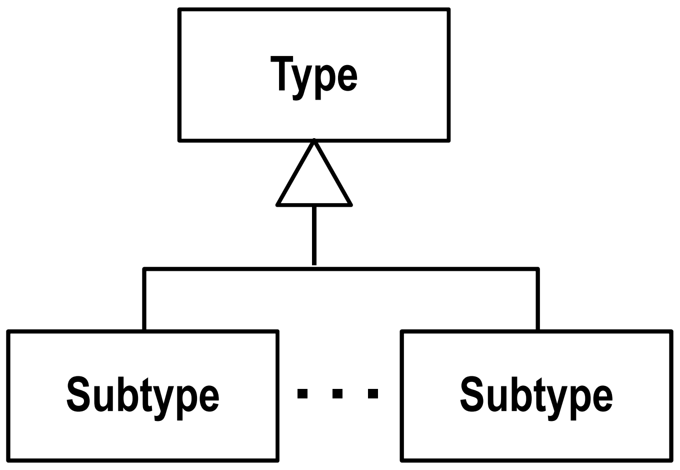
Some RM-ODP GRM Core Relationship Types



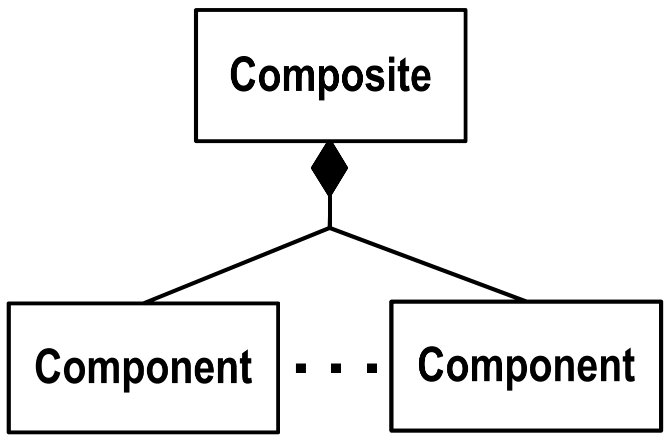
Symmetric Relationships



Dependency Relationships (Asymmetric)



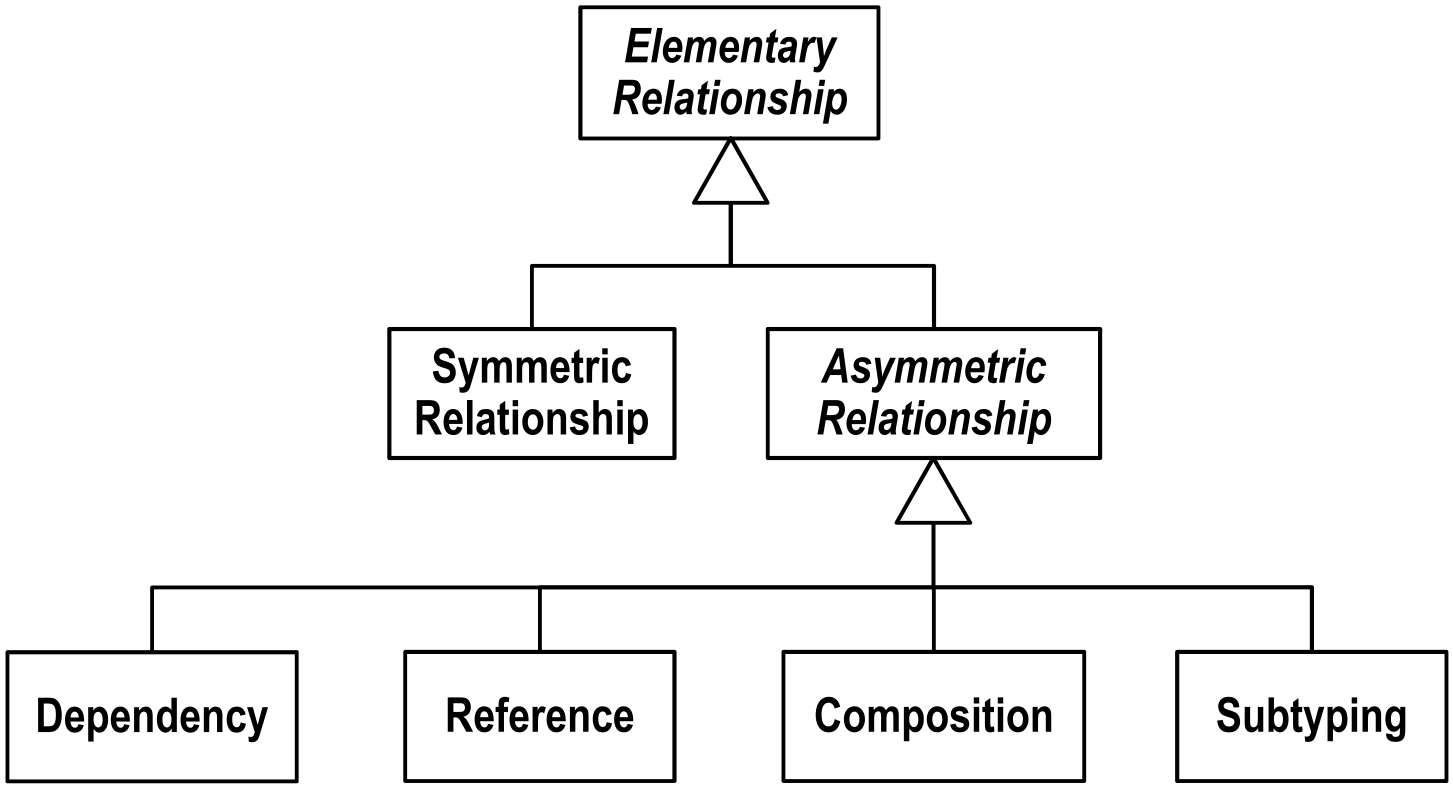
Subtyping Relationships (Asymmetric)



Composition Relationships (Asymmetric)

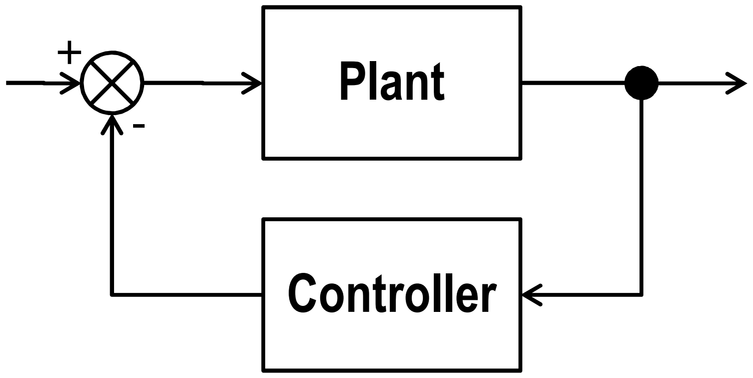
The Kilov Taxonomy

Haim Kilov, "Business Models: A Guide for Business and IT, Prentice Hall, 2002

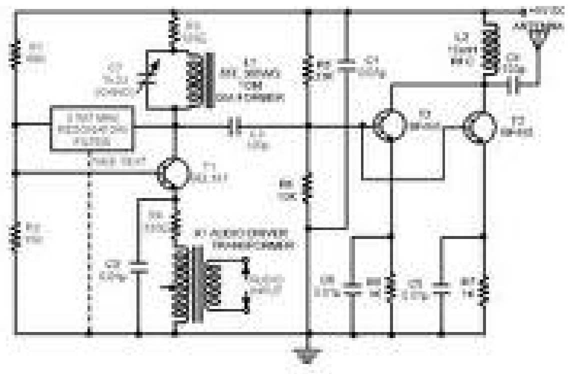


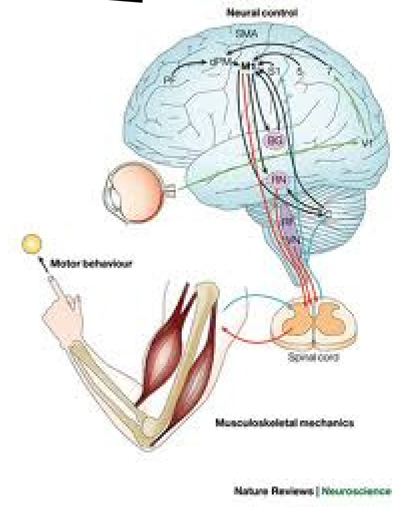
A Crucial Relationship: Abstraction

- The negative feedback control pattern:



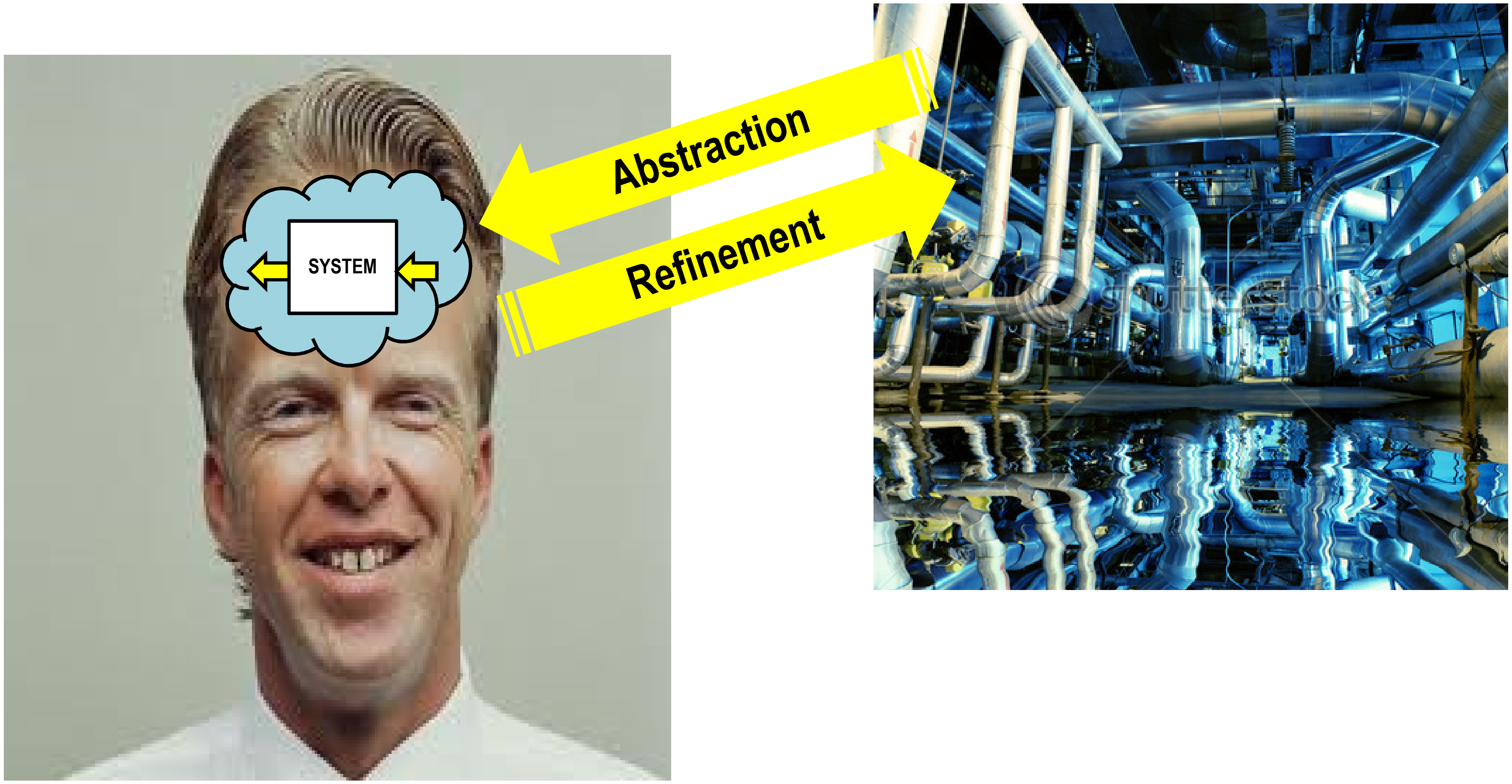
A good abstraction makes the implicit explicit [after Bergson]





Why We Need Abstraction

- An *intellectual defence mechanism* for coping with overwhelming complexity
  - Often, our only mechanism



Abstraction: Definitions

- ABSTRACTION: *A process by which "higher" concepts are derived from the usage and classification of literal ("real" or "concrete") concepts, first principles, and/or other abstractions* [Wikipedia]
- ABSTRACTION (computer science): *a mechanism and practice to reduce and factor out details so that one can focus on a few concepts* [Wikipedia]
- ABSTRACTION: *the act of considering something as a general quality or characteristic, apart from concrete realities, specific objects, or actual instances* [Dictionary.reference.com]

Technical Approaches to Abstraction

- <u>Relaxionist</u>: By loosening constraints
  - Broadens scope of coverage
  - E.g., going from “Square” to “Geometrical Shape”
- <u>Reductionist</u>: By *removing* *or* *<u>absorbing</u>* irrelevant detail
  - (Ir)relevance is a function of viewpoint (set of concerns)

The Big Problem with Abstraction

- **Alfred North Whitehead [*Modes of Thought*]:**

  "The topic of every science is an abstraction from the full concrete happenings of nature. But every abstraction neglects the influx of the factors omitted into the factors retained."

Yikes!! I forgot about the wind and resonance

Fortunately, in engineering, such events are infrequent since the models used in these disciplines are fairly accurate representations of reality

But, When It Comes to Software…

- The day the phones stopped…
  - January 1990; AT&T Long Distance Network crash

```
. . .;
switch (...) {
      case a : ...;
           break;
      case b : ...;
           break;
           . . .
      case m : ...;
      case n : ...;
           . . .
};
```

Recovery time: 1 day

Cost: 100 millions of $'s

"The devil is in the details":

- *Potentially __any line of code can have an enormous impact__*
- *And, there are lots and lots of lines of code*

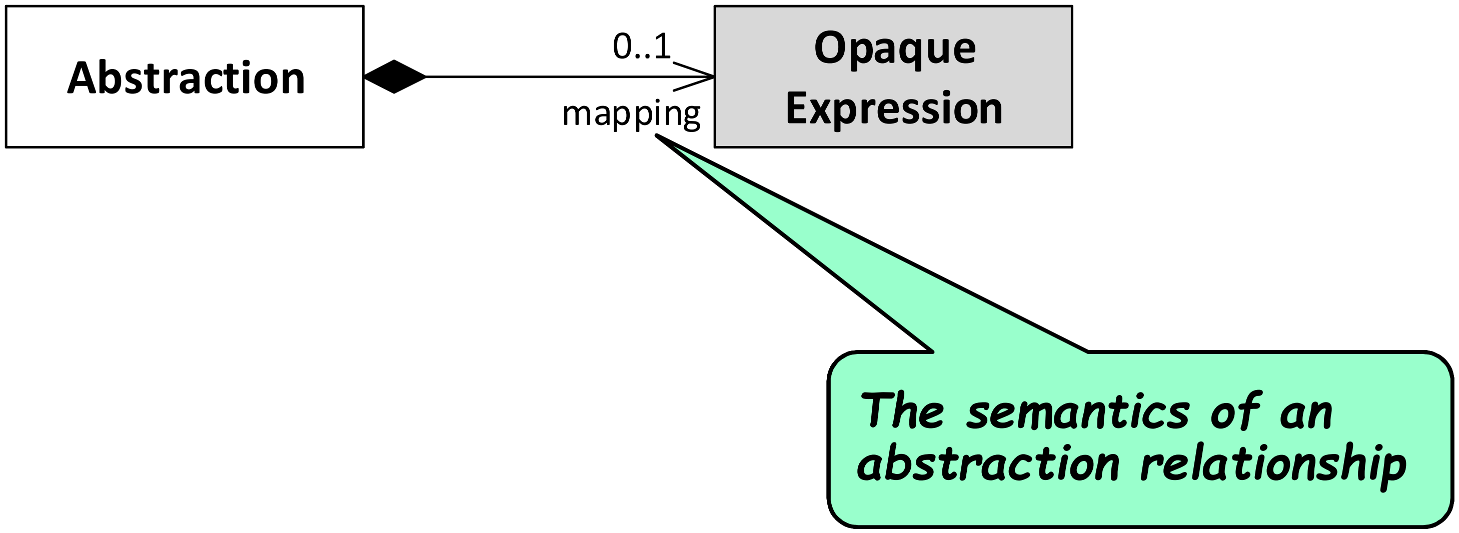
#@$!! I forgot the "break" statement…

So, How Should We Abstract Software?

- Verifying software at the level of individual programming language statements <u>rarely scales</u>
- The only practical way to verify complex software systems is to verify abstract representations (i.e., models) of that software
  - Just like in "real" engineering
- This requires accurate models of software
- But, how can we create abstract representations of systems in which even the minutest of details can have profound consequences?
  - Any detail we leave out could be critical!

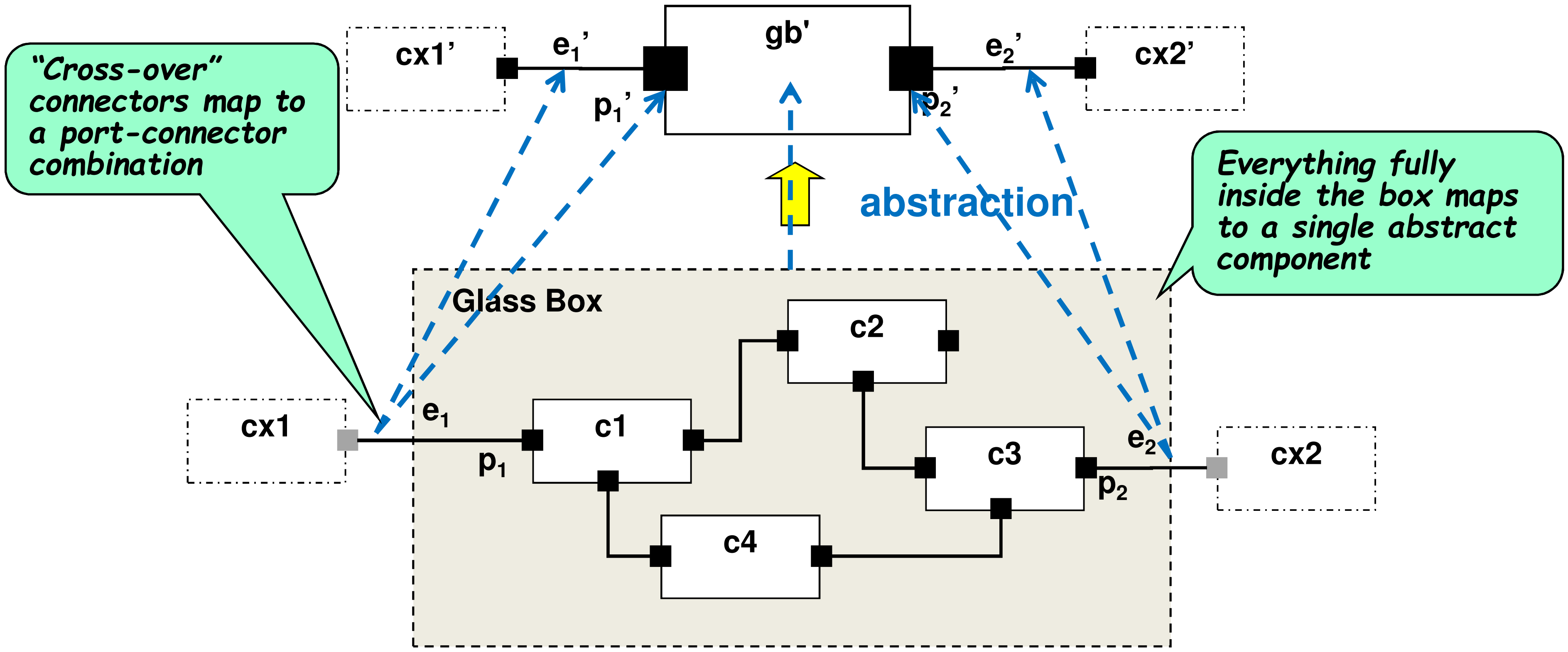
Recommendation

- To avoid such sins of omission, it would be helpful if we are *explicit* about the abstraction process itself
  - i.e., what has been left out, what has been replaced by what, etc.
- Perhaps surprisingly, UML has a (weak) provision for that:



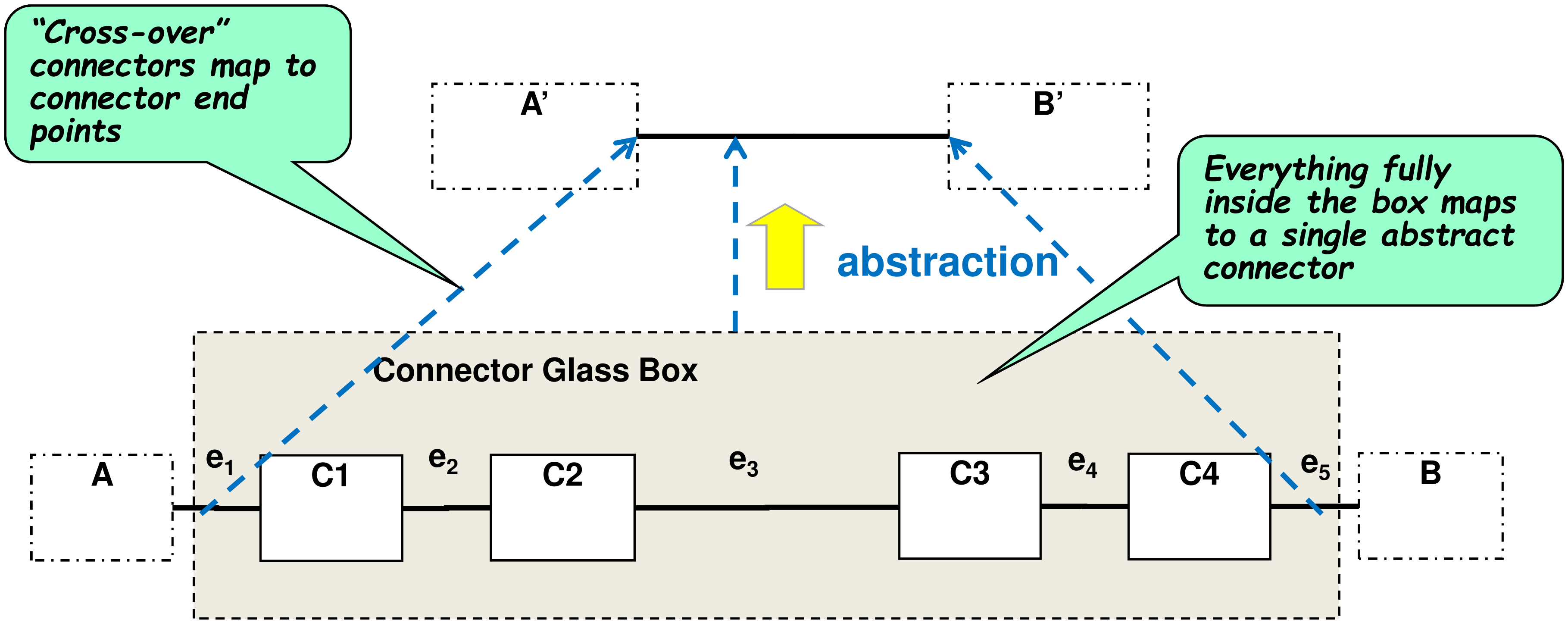
Black Box Abstraction Example

- **Synthesizes a network of tightly-coupled concrete components and renders them as a single high-level component**
  - All mappings from concrete detail to corresponding abstract representation are made explicit


"Cross-over" connectors map to a port-connector combination
cx1'
e_1'
gb'
p_1'
e_2'
p_2'
cx2'
abstraction
Everything fully inside the box maps to a single abstract component
Glass Box
c2
cx1
e_1
c1
p_1
c3
e_2
p_2
cx2
c4

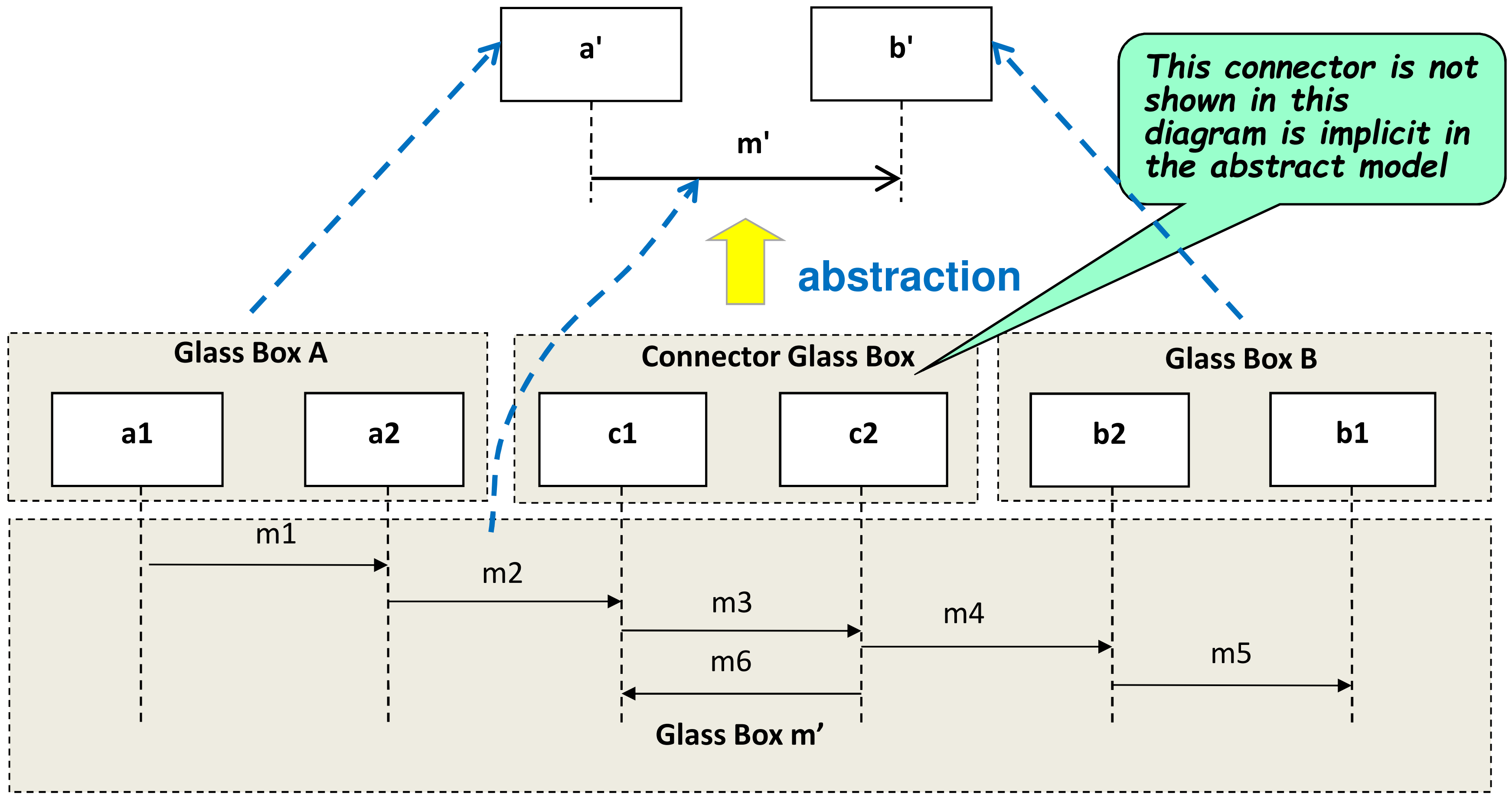
Black Line Abstraction Example

- Abstracts a collection of elements realizing a communications path into a single edge (connector)
  - Could be multipoint


"Cross-over" connectors map to connector end points
A'
B'
abstraction
Everything fully inside the box maps to a single abstract connector
Connector Glass Box
A
e_1
C1
e_2
C2
e_3
C3
e_4
C4
e_5
B

Hybrid Pattern: Layered Communication

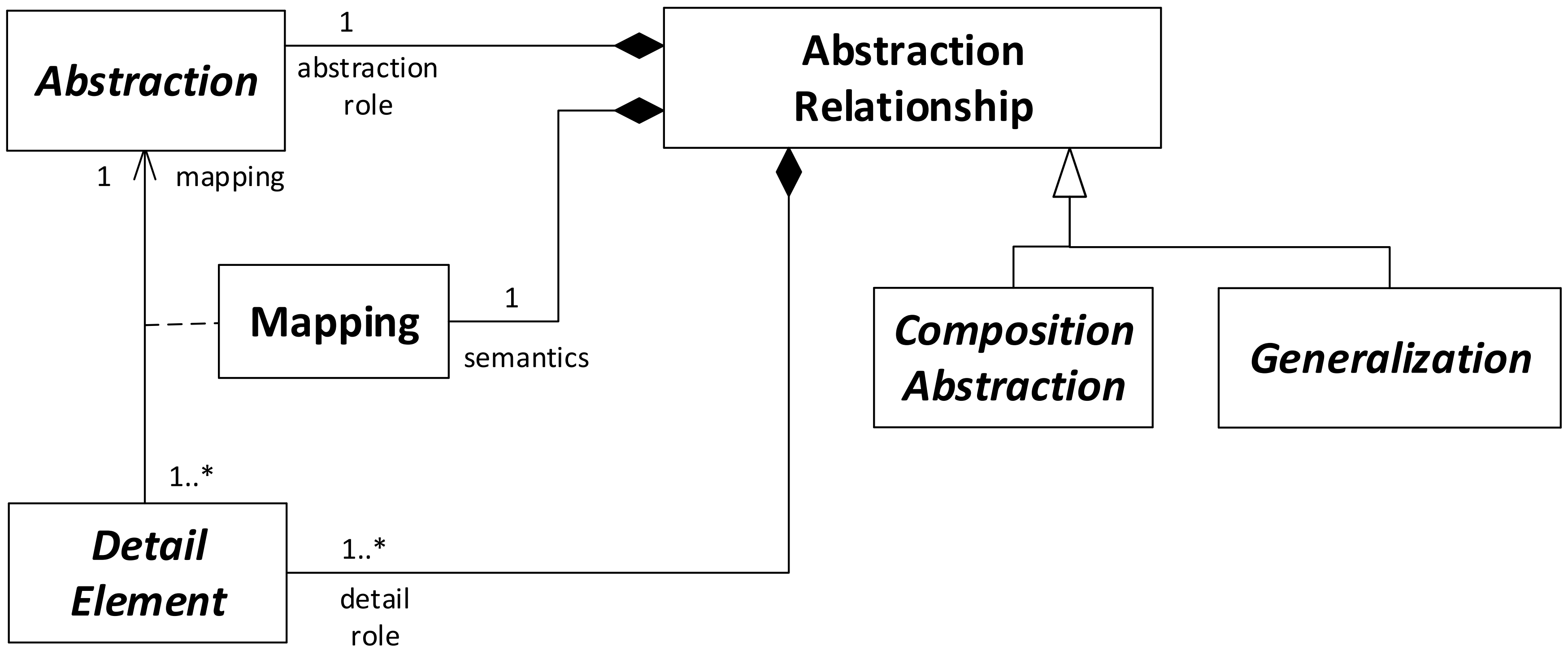
- Combination of structural and behavioural patterns



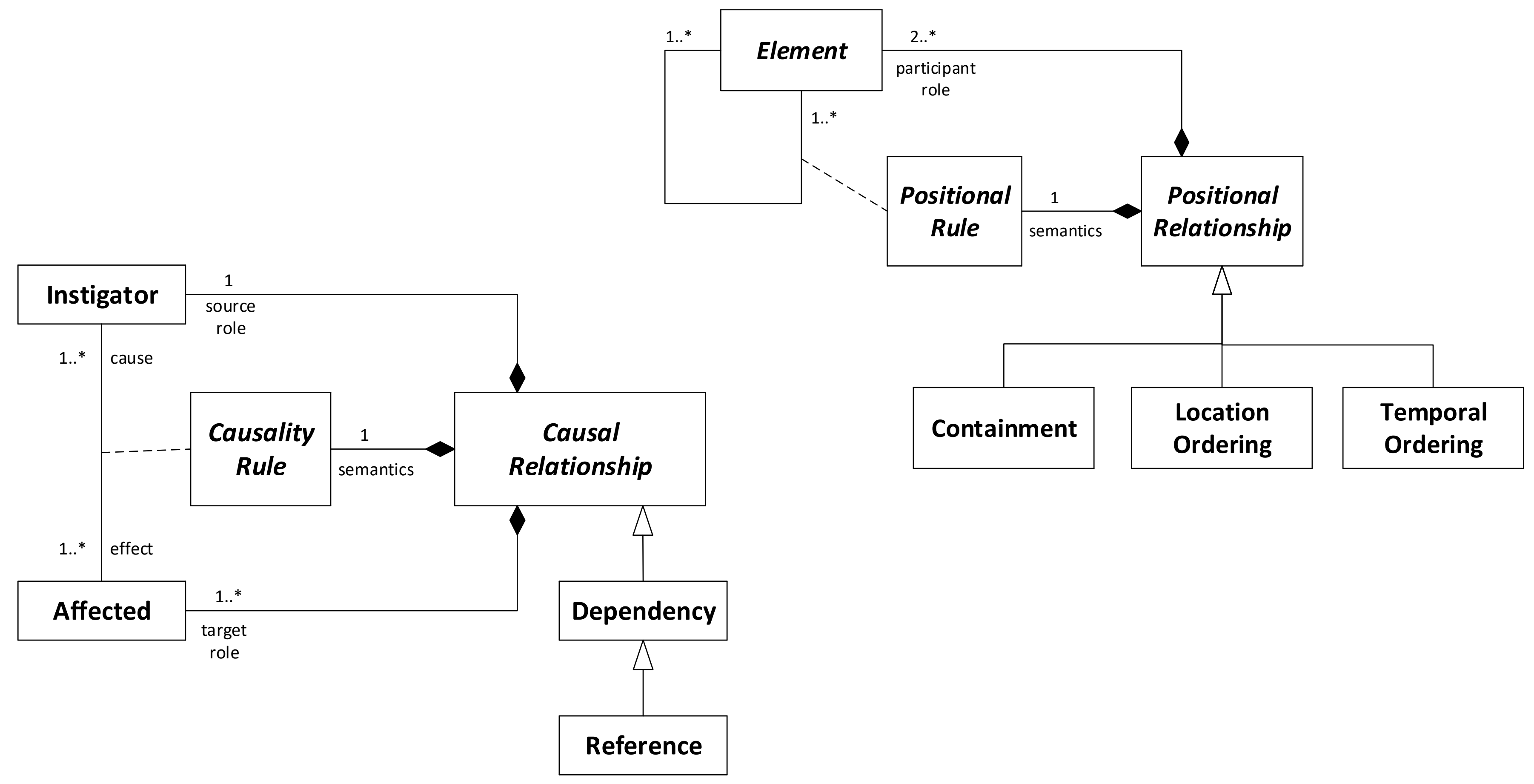
Proposed Abstraction Relationship

- An ***Abstraction*** represents a ***conceptual*** construct:

  "An idea or mental picture of a group or class of objects formed by combining all their aspects." [Oxford dictionary]



Some Other Proposed Core Relationships



Which other core relationships should we define?

Summary

- The relationship concept has been somewhat neglected in recent work on modeling (e.g., UML, SysML)
  - Although there are some excellent precedents in prior work
- In particular, weak ability to specify both machine- and human-comprehensible <u>semantics</u> of relationships
- Most interesting relationships involve dynamic semantics $\Rightarrow$ behavior
- This preliminary work proposes a putative generic model that
  - a) Allows for specifying dynamic as well as structural semantics of relationships
  - b) Provides the ability to manage (control) relationships

In case of disagreement:
VR Tomatoes available to be thrown at the speaker

- THANK YOU-

QUESTIONS, COMMENTS, ARGUMENTS...