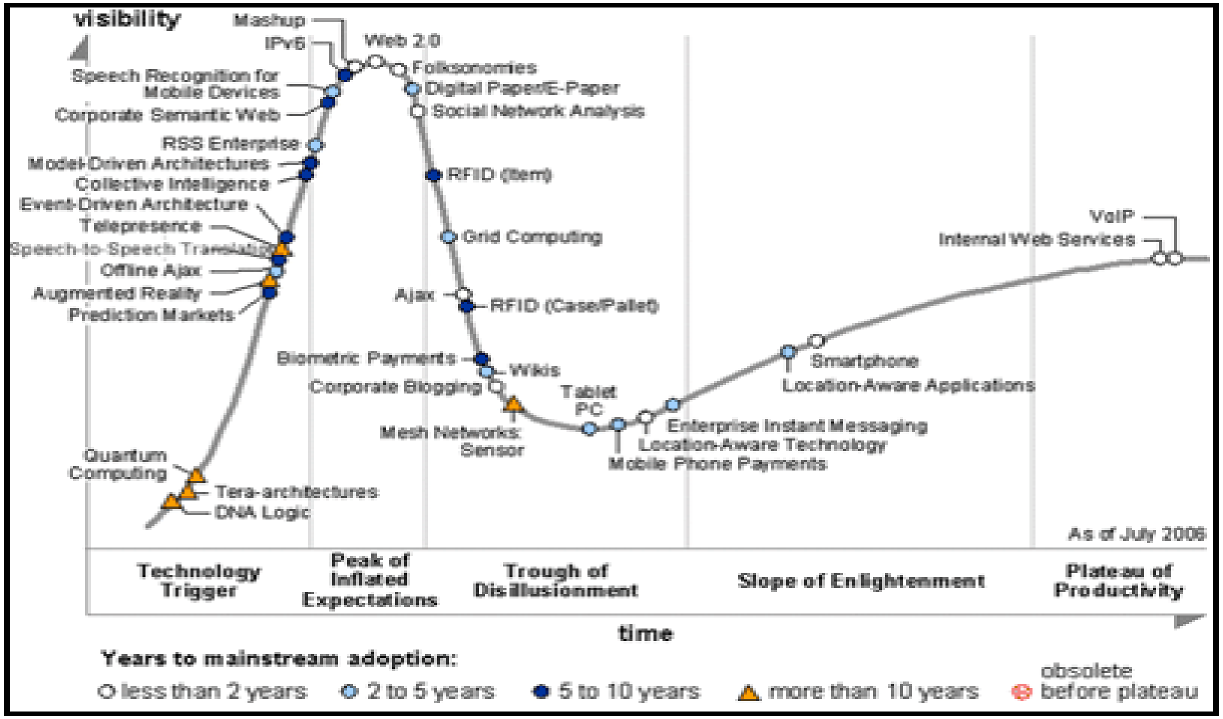
Incremental Semantics for Service Oriented Architecture



Gartner's Emerging Technology Hype Cycle for 2006

Prepared for SOA EGov COP May 2007

Gary Berg-Cross
EM&I
Suite 350 455 Spring park Place
Herndon VA 20170
703-742-0585

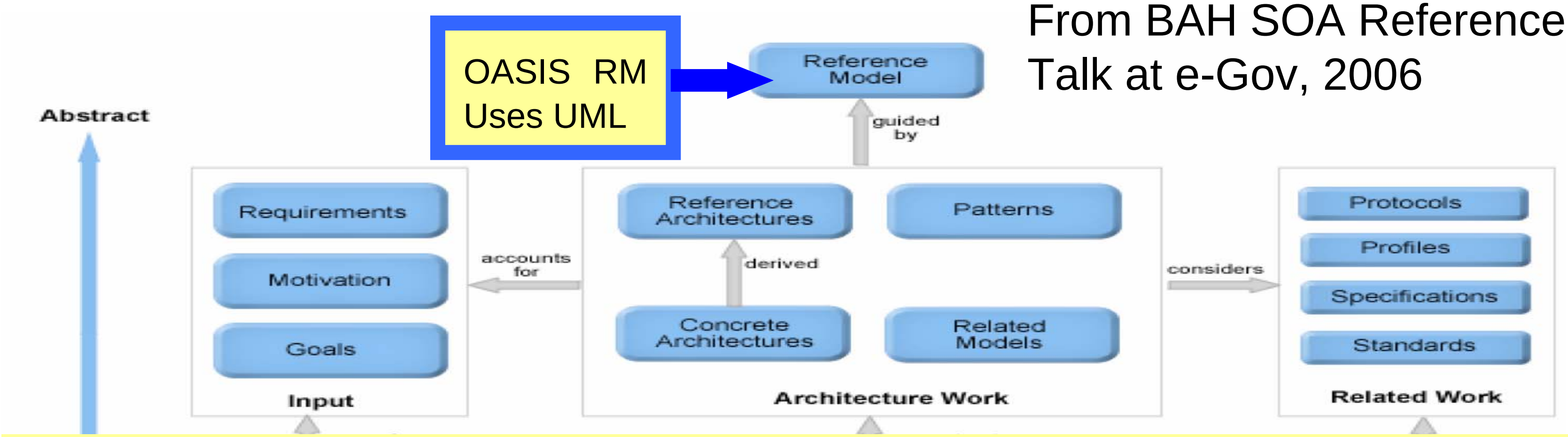
Outline of Discussion

- Background – SOA promises benefits
  - BUT the architecture needs to be well founded
- SOA has been moving (incrementally?) towards grounding in Semantic Architecture Models –”ontologies”
  - But what steps are needed to get there?, what’s the role of enterprise architectures, ontological engineering?
- Semantics are more than a thing (ontology), it is a method, needing an Incremental approach
  - How ontologies are created
  - 4 examples of semantic problems “better” conceptualization, commitment and language representation handles.
- Recap and why is it hard?

SOA Foundations and Benefits

- A core idea is that SOA implementations can be founded on an integrated reference model, reference architectures, standards and specifications

From BAH SOA Reference Model Talk at e-Gov, 2006



OASIS RM
Uses UML

Not just technical and syntactic integration but semantic integration:

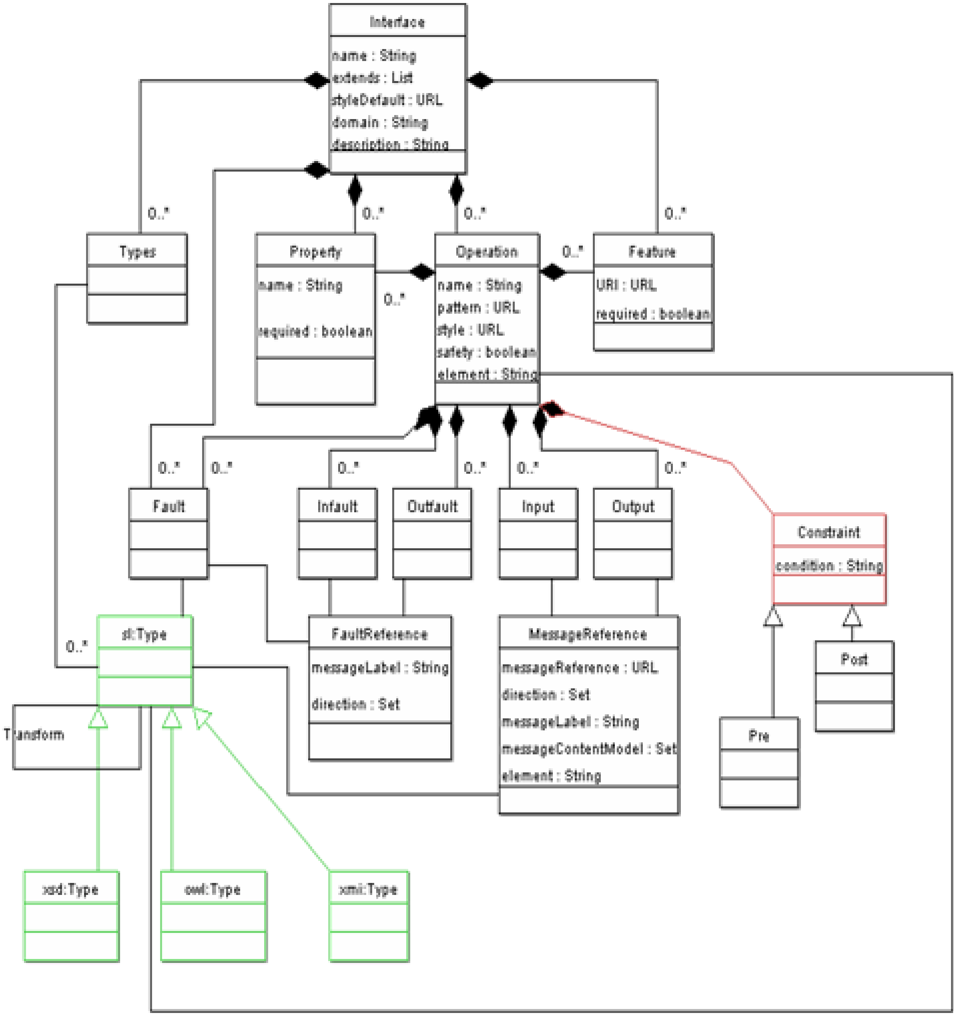
A formal mapping of the meaning of terms from different information sources needs to be built.

This would allow services to move data in and out of “systems” while ensuring that the data is referring to the same thing (or translated into the same).

To do information integration, a “dictionary” must go way beyond simple metadata to deliver meaningful real-time business information.

But SOA Reference Models are often Quite Informal or Lack Content

An Abstract Reference Model for SOA



WSDL-S Meta-model -Names of important Concepts with relations and attributes

Recap: Evolution of “SW”-Objects, Components, Services and Enterprise Architectures

Software Engineering

1. Pre O-O -> Spaghetti code: little explicit structure, no classes
2. Object-Oriented programming provides early roots of SOA
    1. Classes (encapsulation) call each other as services ..but in the same Application
3. With a network, classes are not on the same machine
    - Now a service class find what it needs via an explicit Service Description
    - And a class send its information (passes values) to the other class via XML/XMLS serialization.

Enterprise Architecture

- Start might be IM
- Zachman Framework
- Federal EAs
    - A mix of IM, ERA and BP models
    - Problem - meta-models used to capture architecture are typically **semantically weak**. This criticism goes back to 92 Sowa & Zachman Many EAs are based as much on natural language descriptions as structured models. As a result of the use of conventional IT formalisms, EA models leave implicit many of the details required to understand one architecture and integrate it with others

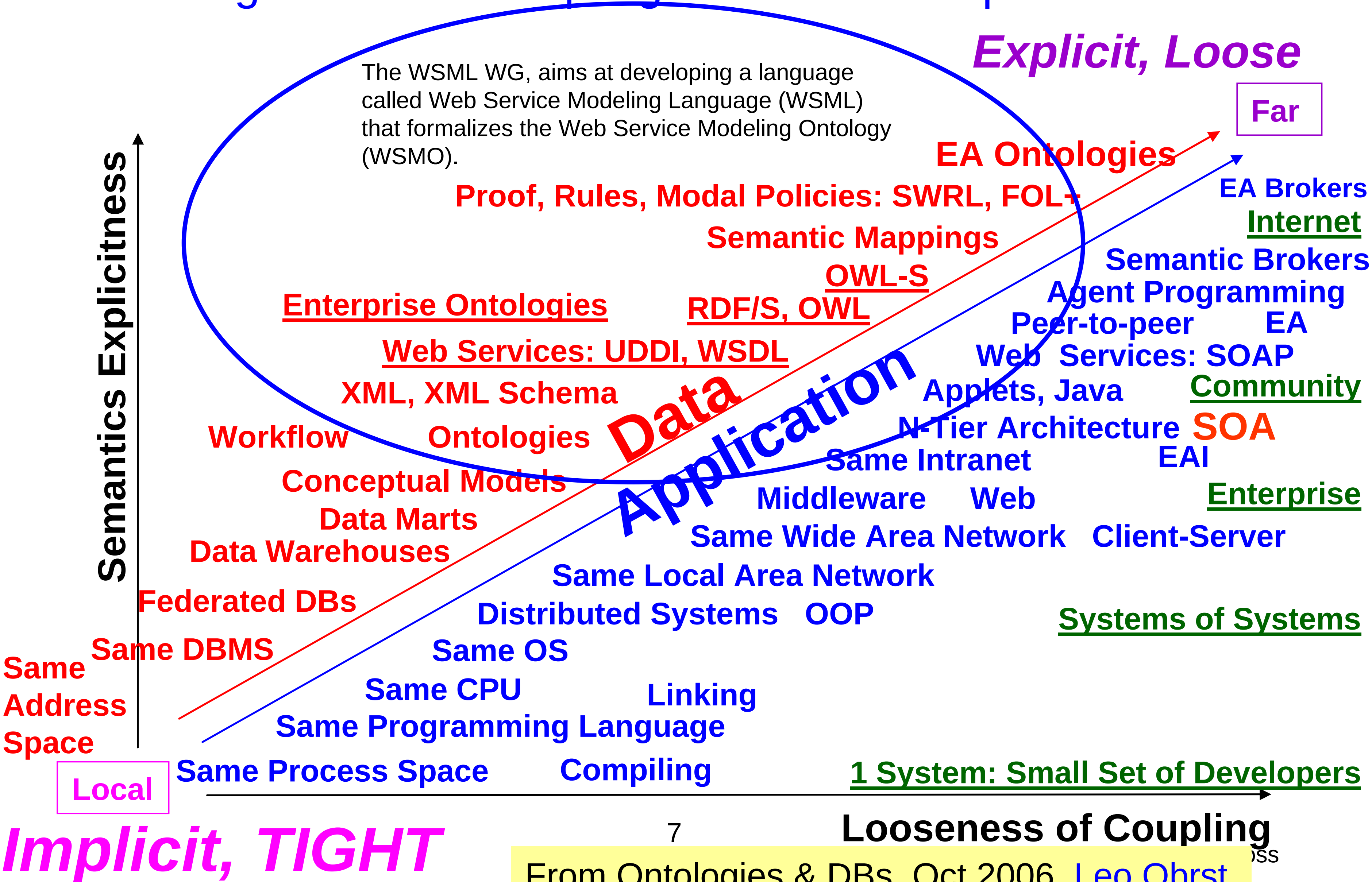
- **Properties of a Final Architecture are clearer than how to get to Semantic Architectures that normalize domains**
- **EAs are moving incrementally towards better semantics (**DRM has added taxonomies for controlled meaning) **but somewhat “piecemeal”**
- **What’s the EA methodology?**

Now, Isn’t it just “Model Driven Architecture”?

(Ed Seidewitz giving tutorial on MDA here today)

- The Object Management Group (OMG), developed Model Driven Architecture™ (MDA™)
- MDA encourages efficient use of system models in the software development process, and it supports reuse of best IT modeling practices when creating families of systems as a way of modeling business process being supported by services.
- Four principles underlie the OMG's view of MDA:
  - Models expressed in a well-defined notation are a cornerstone to understanding systems for enterprise-scale solutions.
  - The building of systems can be organized around a set of models by imposing a series of transformations between models, organized into an architectural framework of layers and transformations.
  - A formal underpinning for describing models in a **set of metamodels facilitates** meaningful integration and transformation among models, and is the basis for automation through tools.
  - Acceptance and broad adoption of this model-based approach requires industry standards to provide openness to consumers, and foster competition among vendors.
- OMG established modeling standards *for* Computation Independent Model (CIM): a model that is independent of computation representations
  - Unified Modeling Language (UML), (a 4-layer metameta-model- their road to semantics?)
  - Meta-Object Facility (MOF), MOF is defined by MOF
  - XML Metadata Interchange (XMI), and
  - Common Warehouse Meta-model (CWM).
- **This is a piece, but just one set of standards in a larger family with varying formality and semantic expressiveness.**

An Established View of This evolution concerns the "Tightness" of Coupling & Semantic Explicitness

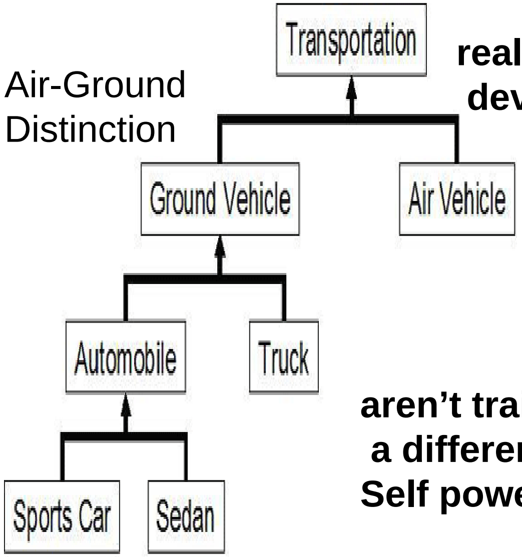


The WSML WG, aims at developing a language called Web Service Modeling Language (WSML) that formalizes the Web Service Modeling Ontology (WSMO).

From Ontologies & DBs, Oct 2006, Leo Obrst

Perhaps the Road Implied by all of These is Not so Direct

As part of the DRM, federal agencies will categorize their data and information assets, as “they deem appropriate and most beneficial to their stakeholders”, in accordance with the elements of an XML schema using taxonomies and topics.
But a problem is illustrated by a sample taxonomy offered as part of DRM 2.0 shown below.



Air-Ground Distinction

really of transport devices

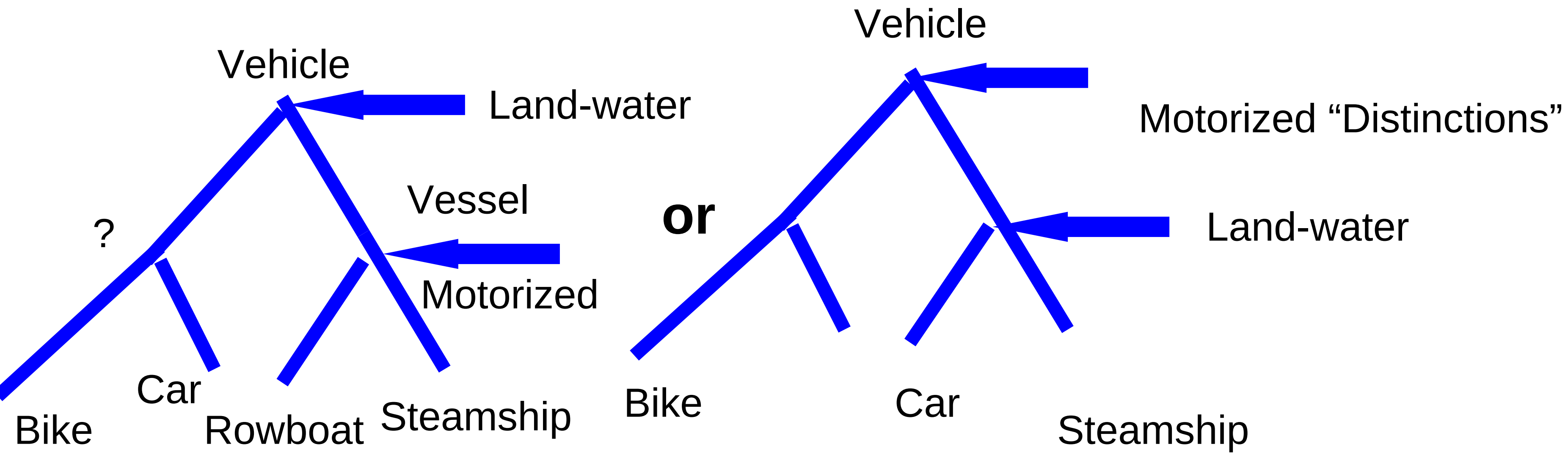
aren’t trains and autos a different sub-type? Self powered?

Bikes, Wheelchairs ?

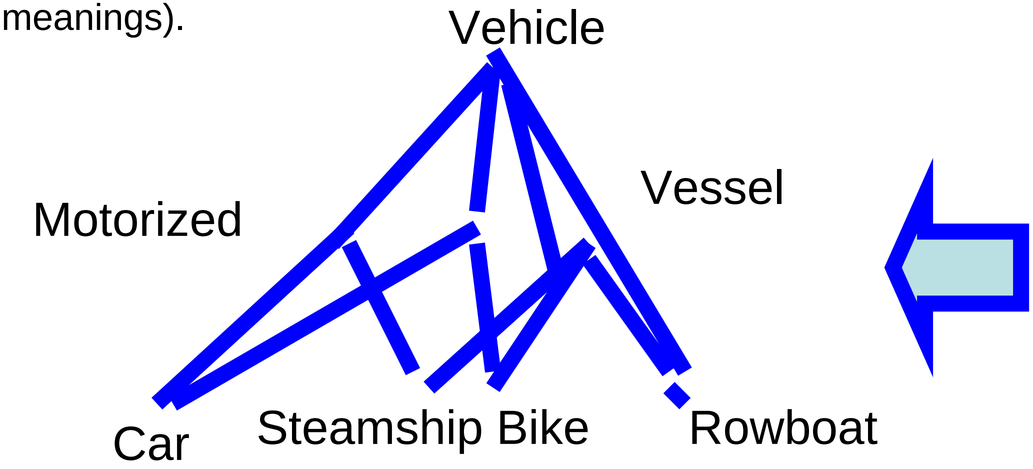
```
<?xml version="1.0"?> <rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"

xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"

xmlns:owl="http://www.w3.org/2002/07/owl#"

xmlns:daml="http://www.daml.org/2001/03/daml+oil#"

xmlns="http://www.owl-ontologies.com/unnamed.owl#"
xmlns:dc="http://purl.org/dc/elements/1.1/"

xml:base="http://www.owl-ontologies.com/unnamed.owl"> <owl:Ontology
rdf:about=""/> <owl:Class rdf:ID="Transportation"/> <owl:Class
rdf:ID="AirVehicle"> <rdfs:subClassOf rdf:resource="#Transportation"/>
Etc.
```

A very informal hierarchy of transportation concepts represents a pseudo-formalization not based on a deep conceptualization and categorization of the domain in terms of distinguishing properties or systematic relations between levels.
This is not an uncommon problem and reflects the lack of the necessary conceptual analysis going Into EAs and Service models

Conceptualizing Taxonomic Structure can be Complex



Distinctions can be applied in any order and this may produce artifacts (like non-motorized air vehicle that is not relevant to intended meanings).

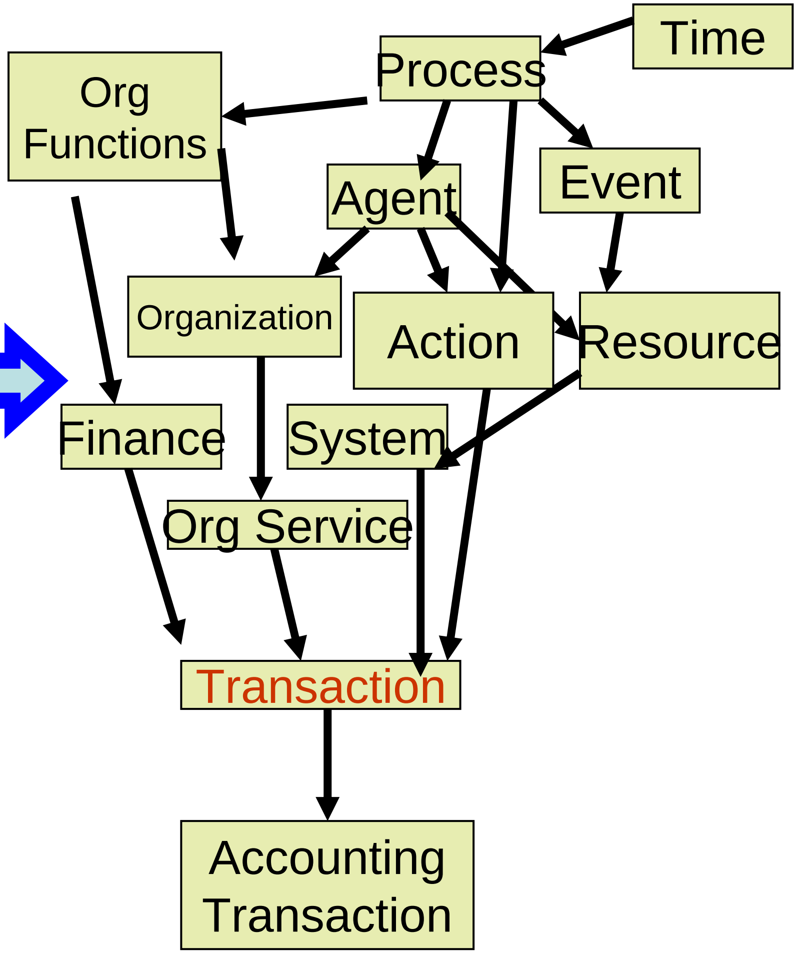


But merging two such hierarchies produces a heterarchy called a semi-lattice, a more complete view

After Andrew Frank's "Distinctions Produce a Taxonomic Lattice, FOIS, 2006

Architectural Problems - EA/SOA Models/Products are very Different from Ontological “Models”

- EA frameworks approach “model levels” very differently than ontologies
- Ontological levels vary based on abstraction and scope:
  - Formal, general and high level concepts that provides names of basic semantics as a basis for understanding of “lower” ontologies such as:
  - Cross domain/enterprise ontologies that describe the scope of an organization which requires integrated concepts etc.
- In contrast, EA tends to be strategic pictures or simple lists at the top so we **can’t ground Service Architectures there**.
- Below the EA Top Level is a “Conceptual” Level, but the formalism for this level might be an ER diagram. Weak semantics. No help there



These reflect different Conceptualizations & analytic methods, not just differences in formalisms.

Incrementally Better “Semantics” in RDFS & OWL

How about Formal Ontologies?

In philosophy *Formal ontology* is defined as “the systematic, formal, axiomatic development of the logic of all forms and modes of being” [1].

In Information Sciences we employ the term *formal ontology* to designate an explicit specification of a shared conceptualization that holds in a particular context. In other words, an ontology provides an explicit conceptualization which describes semantics of data, providing a shared and common understanding of a domain.

1. N. Cocchiarella, *Handbook of Metaphysics and Ontology*,1991

Idea is that we can at least use these terms systematically. **But seems we could say**…

<Human,type,Species> and <Amber,type,Customer>

Very different meanings…..RDFS hasn’t distinguished between classes (Human) and instances (Amber)

RDFS doesn’t have suitable axioms to guide us on this use,

it’s an **incremental step** but remains **too ad hoc**.

How do we **better** specify the intended meaning of this vocabulary and others?

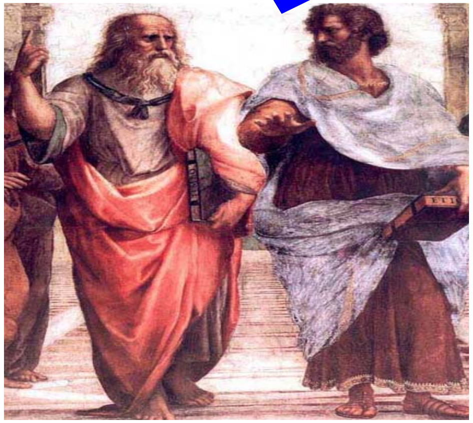
One View of How Ontologies Arise from “Analysis and Conceptualization”

1 World Situations

Interaction

..Bottle on Table. My experience is that “on” is /invariant/

Pragmatic validation



2 Abstraction

Conceptualization

starts to model (part of) the world

UML

OWL

Bottle on Table Intuition expressed in Montague’s semantics for A Language Things D in world state W with conceptual relations R

C= <D, W, R>

To express C Need Language **L (**Terms in L correspond to entities in world) and assign Interpretative Functions I To non-primitive Symbols –a Commitment K.

K=<C,I)

L’s semantics Match Conceptualization

Models for Domain D Expressible In L

Intended Model Fitting C

Ontology Models for D Expressed In L using K

Our Ontology Product

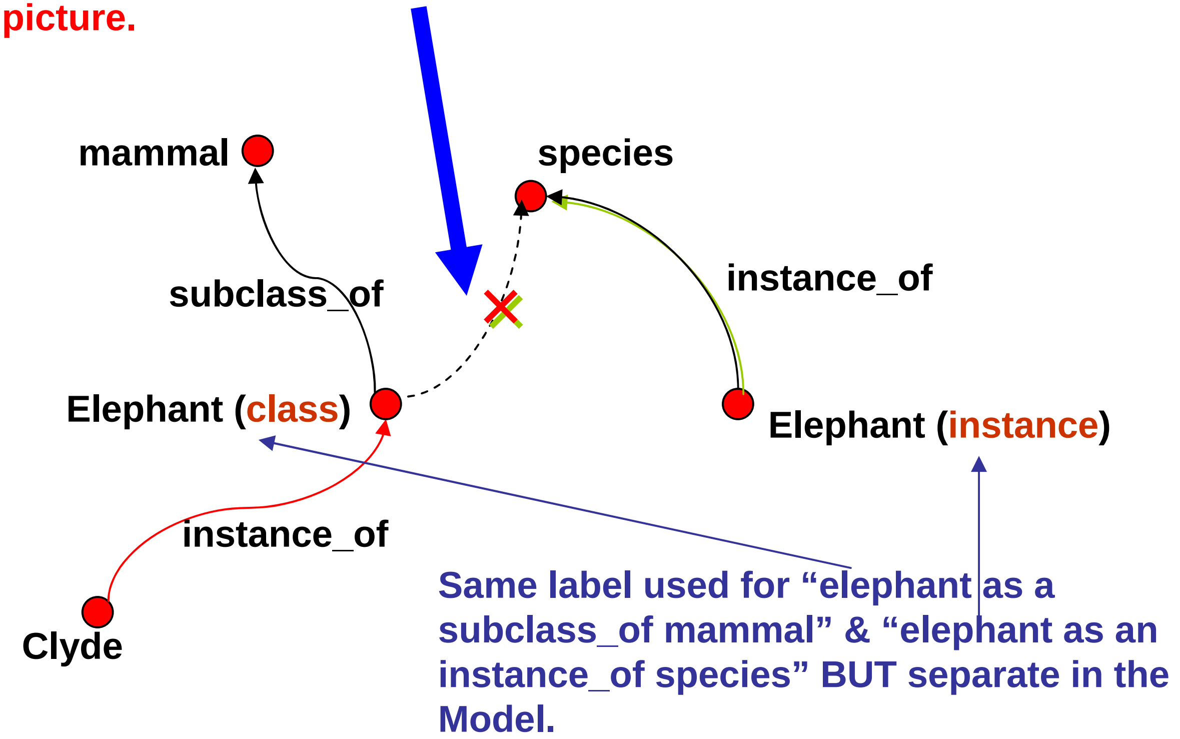
(C for D with K in L using Model)

Adapted liberally from Guarino’s 1998 **Formal Ontology in Information Systems**

Models defines relationship between L syntax and interpretations

Increment 1-Committing to Classes

- Ontology models need to permit a conceptualization of classes to be treated simultaneously as both collections and individuals (instances) which is needed to avoid Transitivity errors as shown in this example from Sowa:
- **Clyde is an elephant.   Elephant is a species.**
- **Therefore, Clyde is a species. Why is this wrong?**
  - **Problem is clear in a portion of an ontology as shown below. A more comprehensive picture.**

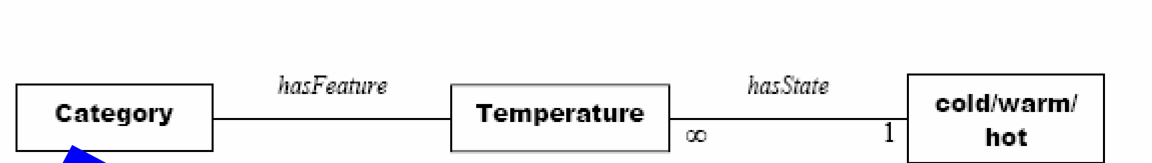


Same label used for "elephant as a subclass_of mammal" & "elephant as an instance_of species" BUT separate in the Model.

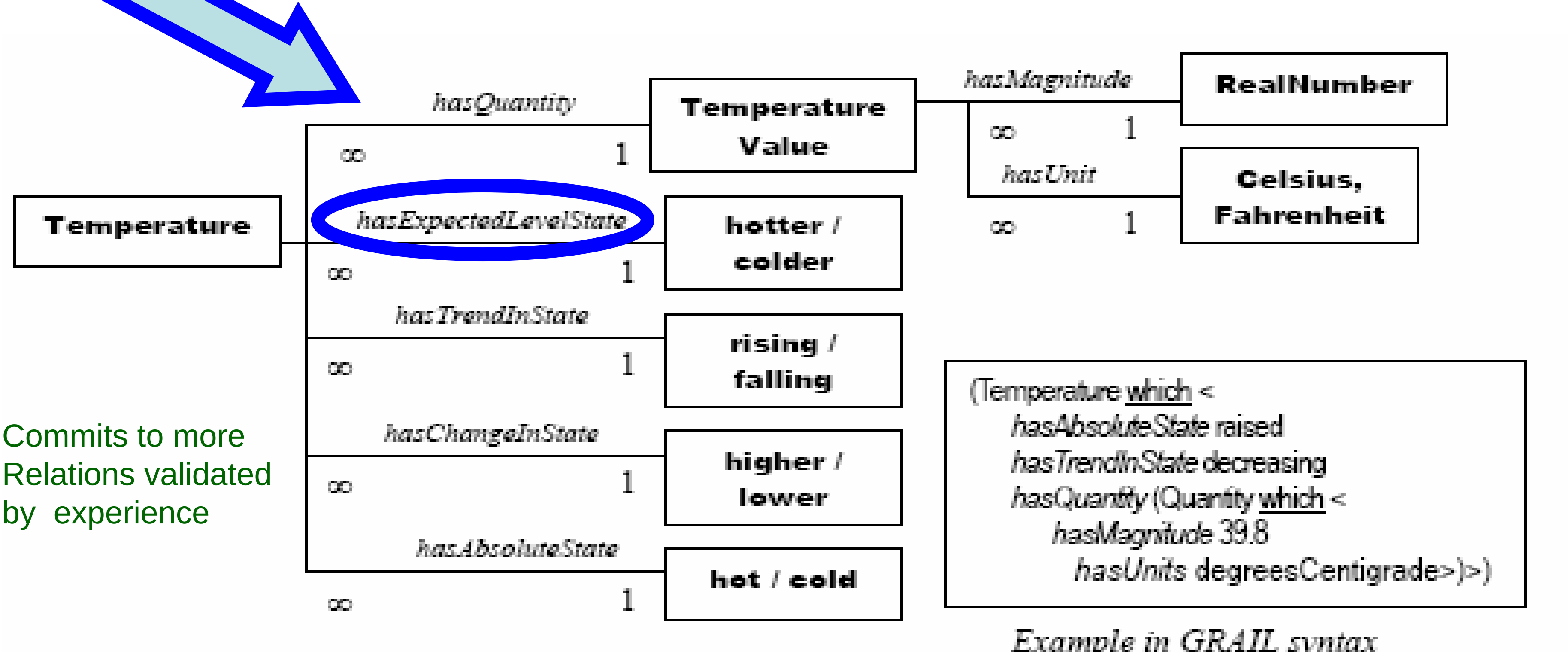
After Leo Obrst "Ontologies and the Semantic Web: An Overview" 2004

John Sowa, 2000. Knowledge Representation: Logical, Philosophical, and Computational Foundations. Pacific Grove, CA: Brooks/Cole Thomson Learning.

Increment 2 : Richer Conceptualization & “Schema”



Simple Feature-State “Model” (from GRAIL) becomes a “richer” schema



(Temperature which <
hasAbsoluteState raised
hasTrendInState decreasing
hasQuantity (Quantity which <
hasMagnitude 39.8
hasUnits degreesCentigrade>)>)

Example in GRAIL syntax

Commits to more Relations validated by experience

In a Language

Increment 3: Better Conceptualization of Part-Whole

- Composition is important to SOA so part-whole relations need to be well founded
- Properties of relations should be better distinguished in EAs and SOAs:
  - Distinguish part by types of entities – physical (finger, hand) or geographic regions (VA, USA)
  - These have relations of: parthood, componenthood (as “functional units”), containment (Asymmetric relation)
    - Amber is part of the SOA group. Amber’s head is part of Amber. Amber’s head is part of the SOA group?
      - Containment is NOT parthood – A group contains Amber.
    - NOT all parts of a whole are meaningful components
      - Amber’s heart has a left side component but does a water drop? No functional parts.

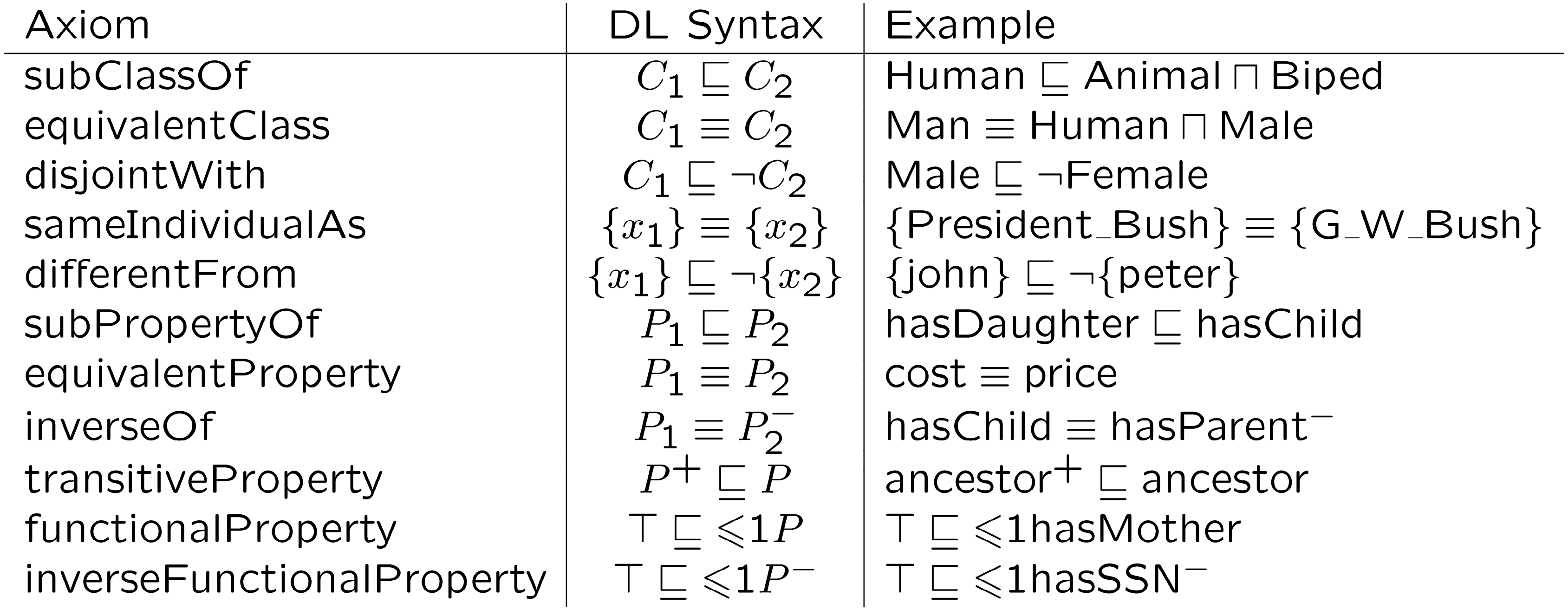
See Odell, J.J. Six different kinds of composition. *Journal of Object Oriented Programming*, 5 (8). 10-15. or http://www.w3.org/2001/sw/BestPractices/OEP/SimplePartWhole/

Increment 4: Adequate Representation - OWL Builds on Layers below it in the Semantic Web stack

The XML syntax for exchange & XML data types (how OWL is expressed)

- RDF instances & RDFS generic (ontology) statements:

- OWL supports mapping among ontologies:
  - Import one ontology into another: all things that are true in the imported ontology will thereby be true in the importing ontology
  - Assert that a class, property, or instance in one ontology/knowledge base is equivalent to one in another ontology



Axiom	DL Syntax	Example
subClassOf	$C_1 \sqsubseteq C_2$	Human $\sqsubseteq$ Animal $\sqcap$ Biped
equivalentClass	$C_1 \equiv C_2$	Man $\equiv$ Human $\sqcap$ Male
disjointWith	$C_1 \sqsubseteq \neg C_2$	Male $\sqsubseteq \neg$Female
sameIndividualAs	$\{x_1\} \equiv \{x_2\}$	{President_Bush} $\equiv$ {G_W_Bush}
differentFrom	$\{x_1\} \sqsubseteq \neg\{x_2\}$	{john} $\sqsubseteq \neg${peter}
subPropertyOf	$P_1 \sqsubseteq P_2$	hasDaughter $\sqsubseteq$ hasChild
equivalentProperty	$P_1 \equiv P_2$	cost $\equiv$ price
inverseOf	$P_1 \equiv P_2^-$	hasChild $\equiv$ hasParent$^-$
transitiveProperty	$P^+ \sqsubseteq P$	ancestor$^+ \sqsubseteq$ ancestor
functionalProperty	$\top \sqsubseteq \leqslant 1 P$	$\top \sqsubseteq \leqslant$1hasMother
inverseFunctionalProperty	$\top \sqsubseteq \leqslant 1 P^-$	$\top \sqsubseteq \leqslant$1hasSSN$^-$

From 2004 Tutorial on OWL by
Peter Patel-Schneider

Simple Goals for a Quality Ontology for SOA

– An ontology results from ontological engineering and the resulting product should be :

1. Correct/valid - captured intuitions of domain experts
2. Meaningful - all named classes can have instances
    1. Heterarchy example to aid merging of taxonomies
3. Rigorous – stands up to rational analysis
    1. Such problems as when we simultaneously say that a financial process is caused by one or more ordered assemblies of business functions, and that view financial processes as decomposed from business functions using *part-of relations*.
4. Minimally redundant - no unintended synonyms/terms
    1. Are asset and resource the same or is one a sub-type of the other?
5. Sufficiently axiomatized – include detailed constraining descriptions as axioms
    1. E.g. if event e1 has a causal influence on event e2, then e1 must precede e2 in time.
6. Formal –can be represented/put into a form amenable to automated processing

Recap and Methodology Enhancement

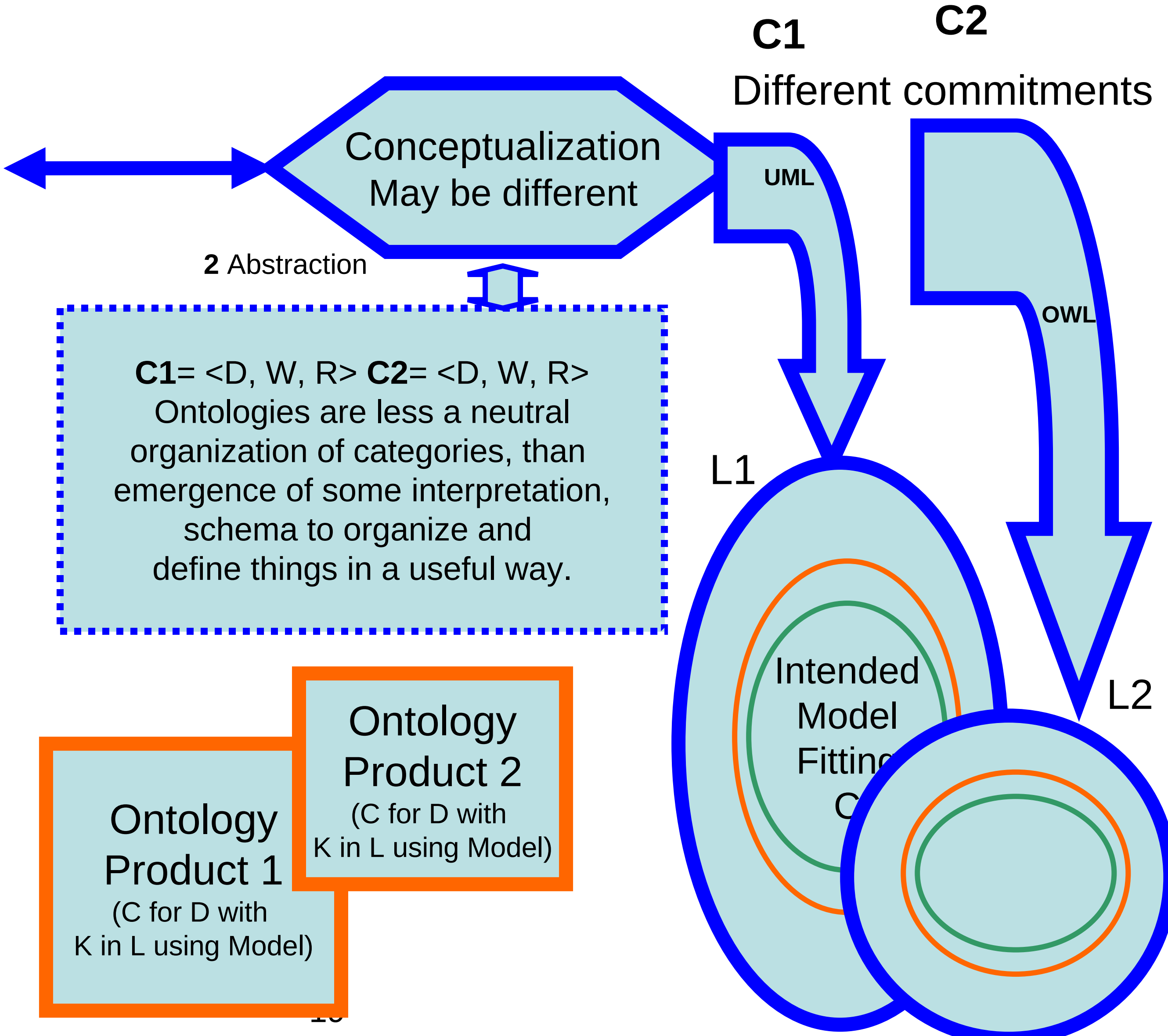
- We need to stress improvements to conceptual analysis and rational commitments in our models. E.G.
  - How to collect general terms describing classes and relations to be employed in the description of a domain;
  - Organize the terms into a taxonomy of the classes by the ISA relation; merging these etc. and
  - Expressing these in an explicit way with constraints that make these classes/terms usefully meaningful.
- These in turn need to be faithfully formalized in ontological languages that can express the intended semantics.
- We need a balanced approach across the ontological development process.

Why is it Hard? Merging requires commonality

Modelers &
Model

- Our SOA methods inherit too much from semantically weak methods constrained by formalisms.

- Better formalisms are available, but unless semantic analysts match these with suitable analysis and design methods we won't get the semantics we need.



C1= <D, W, R> **C2**= <D, W, R>
Ontologies are less a neutral organization of categories, than emergence of some interpretation, schema to organize and define things in a useful way.

Ontology Product 2
(C for D with K in L using Model)

Ontology Product 1
(C for D with K in L using Model)

Why it is Hard? Some Additional Thoughts

- Ontology is more than a thing, it is products that arises by methods which needs several things coordinated. E.g.
  - Rigorous and “Abstract” analysis & design
    - an old topic in Software Engineering that Applies to EA & SOA but there are too few implementations of these ideas
  - Balanced Semantic Analysis, aligned to formalisms, is needed to Develop Adequate Service Models
- Some of the increments have been illustrated to overcome typical errors, but many more could be cited.
- A barrier to “better” SOA Semantics is the lack of an off the shelf ontological engineering method for SOA
  - This is hard due to the scope needed, the lack of expertise among SOA workers and the nature of the work which combines the scruffy as well as the neat.

Some Sources

J. F. Sowa, *Knowledge Representation. Logical, Philosophical and Computational Foundations*, Brooks/Cole, (2000).

Handbook on Ontologies Series: International Handbooks on Information Systems
Staab, Steffen; Studer, Rudi (Eds.) 2004, XVI, 660 p., 190 illus., Hardcover
ISBN: 978-3-540-40834-5

Ontological Engineering: with examples from the areas of Knowledge Management, e-Commerce and the Semantic Web. First Edition by Asuncion Gomez-Perez (Author), Oscar Corcho (Author), Mariano Fernandez-Lopez

FOIS-2006, International Conference on Formal Ontology in Information Systems, Bennett and Feldman (eds.) IOS Press Includes Nontological Engineering by Waclaw Kusnierczyk, example of a systematic attempt to define ontology.