Logical Modeling for Engineering

Conrad Bock
U.S. National Institute of Standards and Technology
November 28, 2011
Overview

- Quantitative and logical Modeling
- Categories
  - As membership conditions
  - Terminology and notation
  - Kinds of conditions
  - Most common condition: Generalization
- Categories of categories
  - Subject-specific languages
- Categories of relations (and processes)
- Summary and References
Quantitative Modeling

- Quantitative modeling
  - Numerical formulas (equations)
  - Dynamic and stochastic simulations

- Used for:
  - Calculating or simulating numeric values and probabilities.
  - Deriving new numerical formulas.
Logical Modeling

- Logical modeling is about categorizing things and relations between things ...
  - This document is a requirement, this other one is a design, and the second satisfies the first.

- ... and keeping these categorizations consistent.
  - Requirements or designs are changed, does the satisfies relation still hold?
  - If not, what would make it hold again?
Categories = Conditions

- Things “fall into” categories.
- Categories have conditions for what can and cannot fall into the category.

**ThingsThatFloat**

\{ x : \text{density}(x) < \text{density(water)} \}
Categories **Specify Sets**

- Which things fall into a category can change over time without changing the category (condition).
  - New things created, some things destroyed, conditions met or unmet over time.
  - Not true for set membership.

ThingsThatFloat

\{ x : \text{density}(x) < \text{density(water)} \land \exists x \}\}
Categories Imply Existence

- Conditions might only be satisfied by things from the past, future, in simulation, or not at all.

Roman Cities
- Existed in the past

 Marketable Solar Cars
- Might exist in the future, or in simulation

Perpetual Motion Machines
- Imaginary, or will never exist
Fixed and Changing Conditions

- When something does not fall into a category, but it should, you can:
  - Modify the thing
  - Modify the category’s conditions

Conditions for “as-built” categories are modified if they are not satisfied by something they should be.
Terminology

- The term “categories” is just for explanation in this presentation.
- Other terms:
  - “Classes” in the Unified Modeling Language (UML) and Web Ontology Language (OWL).
  - “Blocks” in the Systems Modeling Language (SysML), an extension of UML.
- Things falling into categories:
  - UML / SysML: “Instances” (of classes / blocks)
  - OWL: “Objects” and “Data” (interpretations of classes and datatypes)
Graphical Notation

UML & SysML:

- ThingsThatFloat
  - UML Class (or SysML Block without stereotype)
  - SysML Block (= UML Class with the Block stereotype applied)

- «block» ThingsThatFloat
  - UML / SysML Constraint:
    - density(self) < density(water)

- "self" variable = any one thing falling into the category

SysML:

- «block» ThingsThatFloat
  - SysML compartment notation for UML / SysML Constraint:
    - constraints
    - density(self) < density(water)

Note: Naming conventions are usually singular, easily confused with instances.
Diagrams

- UML Classes and SysML Blocks appear in particular kinds of diagrams.

**UML Package Diagrams**

```plaintext
package Float

ThingsThatFloat

density(self) < density(water)
```

**SysML Block Definition Diagrams**

```plaintext
bdd Float

«block»

ThingsThatFloat

constraints

density(self) < density(water)
```
“In” Conditions

- Can only determine when something falls into a category, not out of it.
  - Any four-wheeled thing over 750kg that carries people using its own power over 100kw is a car.
  - If something meets the condition it is a car.
  - Otherwise, it might be a car or not (maybe some cars are three-wheeled).

- Purely sufficient conditions do not interact.
  - Each condition is sufficient separately.
“Out” Conditions

- Can only determine when something falls out of a category, not in it.
  - Cars are vehicles.
  - If condition is not met (something is not a vehicle), then it is not a car.
  - Otherwise, it might or might not be a car (some vehicles might not be cars).

- Purely necessary conditions do not interact.
  - Each condition negates separately.

- Combining necessary and sufficient can be contradictory.
In vs Out in English

- Sufficient (in) conditions usually have the category at the end
- Necessary (out) conditions usually have the category at the beginning

Any four-wheeled thing over 750kg that carries people with its own power over 100kw is a car. (sufficient / in)

Cars are vehicles. (necessary / out)

Sometimes sufficient conditions are incorrectly read as also necessary.
In & Out Conditions

- Conditions can be both sufficient (in) and necessary (out).
  - Things must meet the condition to be in the category, otherwise they are out, no in-between.

- **Mathematical set descriptors:**
  - \{ x : \text{density}(x) < \text{density}(water) \}
  - Things less dense than water are in the category (sufficient / in).
  - Things more dense or the same density are not in the category (necessary / out).
The Most Common Condition

- Things falling into one category always fall into another.
  - Example: Cars are vehicles.
  - Cars satisfy the conditions of Vehicles.
  - A necessary condition for Cars, a sufficient condition for Vehicles.
Terminology and Notation

- The previous condition is so common it is given a name and notation in most languages.
- “Generalization” in UML and SysML.
- “Subclass” in OWL.

![SysML/UML Generalization Diagram]
Multiple Generalization

- Useful for reusing and combining categories.
Multiple Generalization Gotchas

- Subcategories might not be complete.
- Subcategories might partially overlap.
- Subcategories might not be an intersection.
Categories in Product Lifecycles

- In the ideal world:
  - Cars as built and maintained are also cars as designed.
  - Cars as designed are also cars meeting requirements.

Conditions are requirements

Cars As Required

Cars As Designed

Cars As Built

Cars As Maintained

Conditions are designs

Conditions reflect what is built

Conditions reflect results of maintenance
Analysis / Reasoning

- In the real world sometimes:
  - Designs do not meet requirements.
  - Cars are not built or maintained to designs.

- Analyzers and reasoners help detect the possibility of these cases earlier.
Categories of Categories

- Distinguish categories according to purpose.

Categories of Categories

<table>
<thead>
<tr>
<th>Requirement Categories</th>
<th>Design Categories</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cars As Required</td>
<td>Cars As Designed</td>
</tr>
<tr>
<td>Trucks As Required</td>
<td>Trucks As Designed</td>
</tr>
<tr>
<td>Planes As Required</td>
<td>Planes As Designed</td>
</tr>
</tbody>
</table>

Requirement Categories (conditions are requirements)

Design Categories (conditions are designs)
Subject-Specific Languages

- Use terminology of subject matter experts, rather than logic / ontology.

Categories

- Requirements
- Designs

Subject specific terminology

Requirements

- Car Requirement
- Plane Requirement
- Truck Requirement

Designs

- Car Design
- Plane Design
- Truck Design
Terminology

- The terms “categories of categories” and “subject-specific languages” are just for explanation in this presentation.

- Other terms:
  - “Metaclasses” in UML/SysML (part of “metamodeling”).
  - “Domain-specific languages” (common in UML community).
  - Not mentioned much in the OWL community, but it is partially supported with “punning”.
Relations

- Relations between actual things or categories:
  - Cars have engines.
  - Designs meet requirements.

![Diagram showing relationships between cars and engines](image)
Terminology

- The term “relations” is just for explanation in this presentation.
- Other terms:
  - “Properties” in UML, SysML, and OWL.
  - “Associations” in UML
- Things falling into relation categories:
  - UML / SysML: “Links” (of associations). No term for properties, but properties have “values”.
  - OWL: Elements of set cartesian (cross) products (“pairs”, “tuples”).
Graphical Notation

UML & SysML:

SysML applies the «block» stereotype to UML Classes.
Generalization of Relations

- Links falling into one relation category always fall into another.
  - Example: Car-engine links are physical containment links.
Terminology and Notation

- The term “generalization of relations” is just for explanation in this presentation.
- Other terms:
  - “Property Subsetting” or “Association Generalization” in UML and SysML.
  - “Subproperties” in OWL.
Subject-Specific Languages

- Use terminology of subject matter experts, rather than logic / ontology.

SysML terminology & notation
## Other Logical Constructs

<table>
<thead>
<tr>
<th>UML/SysML</th>
<th>OWL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Property / Association Multiplicity</td>
<td>Property Cardinality</td>
</tr>
<tr>
<td>Property Redefinition</td>
<td>Property Restriction</td>
</tr>
<tr>
<td>UML Composite Structure, SysML Internal Block Diagram</td>
<td>Role Composition</td>
</tr>
<tr>
<td>SysML Association Participant Properties and Internal Block Connector Properties</td>
<td></td>
</tr>
</tbody>
</table>
Logical Process Modeling

- Categorizing occurrences.

- ChangeColor #1
  - Paint → Dry
  - Falls only into #1

- ChangeColor #3
  - Spray Paint → Dry → Cleanup
  - Falls into both
  - Falls only into #3
Summary

- Logical modeling is about categories.

  - Categories = conditions specifying sets.
    - Independent of things falling into them.
    - In/out (sufficient/necessary) conditions.
    - Common condition: Generalization.

- Relation as categories of links.

- Categories of categories to define subject-specific languages.

- Various terminologies and notations.

- Applicable to product and process modeling.
References

- **UML**: [http://omg.org/spec/UML](http://omg.org/spec/UML)
- **SysML**: [http://omg.org/spec/SysML](http://omg.org/spec/SysML)
- **OWL**: [http://w3.org/TR/owl2-overview](http://w3.org/TR/owl2-overview)
- Other material: conrad dot bock at nist dot gov.