# a separation of concerns

<table>
<thead>
<tr>
<th><strong>semantics</strong></th>
<th><strong>representation</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>what things are there?</td>
<td>what information is stored?</td>
</tr>
<tr>
<td>how are things classified?</td>
<td>which tables or classes?</td>
</tr>
<tr>
<td>how are things related?</td>
<td>what are the datatypes?</td>
</tr>
<tr>
<td>how are things named?</td>
<td>what columns? associations?</td>
</tr>
<tr>
<td>what’s mutable?</td>
<td>what are the keys?</td>
</tr>
<tr>
<td>what updates happen?</td>
<td>encoding generalizations?</td>
</tr>
<tr>
<td>what queries are asked?</td>
<td>optimize common queries?</td>
</tr>
<tr>
<td>what invariants hold?</td>
<td>integrity constraints?</td>
</tr>
</tbody>
</table>
how to avoid semantic models

option #1: don’t articulate semantic model at all
› no basis for design discussion with others
› can’t outsource

option #2: use code to express semantics
› hard to avoid implementation details
› not a lightweight notation

option #3: sketch it out informally
› hard to be precise
› no basis for communication with others, or tools

option #4: write it out in English
› more work, less precise
› hard to see omissions
origins of our OM notation

- logic diagrams (Euler, Venn, Peirce)
- ZF set theory
- relational calculus (Tarski)
- relational model (Codd)
- Z notation
- model checking
- Alloy Language
- ER & other data models
- object model notations (OMT etc)
- Unified Modeling Language
- Alloy Diagrams

1700
1900
1940
1970
1980
1990
2000

mathematical logic
object-oriented development
software verification
relational databases
**alloy**: a language & tool for relational models

**about alloy**

Alloy is a language for describing structures and a tool for exploring them. It has been used in a wide range of applications from finding holes in security mechanisms to designing telephone switching networks.

An Alloy model is a collection of constraints that describes (implicitly) a set of structures, for example: all the possible security configurations of a web application, or all the possible topologies of a switching network. Alloy's tool, the **Alloy Analyzer**, is a solver that takes the constraints of a model and finds structures that satisfy them. It can be used both to explore the model by generating sample structures, and to check properties of the model by generating counterexamples. Structures are displayed graphically, and their appearance can be customized for the domain at hand.

At its core, the Alloy language is a simple but expressive logic based on the notion of relations, and was inspired by the Z specification language and Tarski's relational calculus. Alloy's syntax is designed to make it easy to build models incrementally, and was influenced by modeling languages (such as the object models of OMT and UML). Novel features of Alloy include a rich subtype facility for factoring out common features and a uniform and powerful syntax for navigation expressions.

The Alloy Analyzer works by reduction to SAT. Version 4 was a complete rewrite that included **Kodkod**, a new model finding engine that optimizes the reduction, and a new front end.
why not UML?

what is UML?
› Unified Modeling Language
› collection of many notations
› attempt to merge into standard

what’s good about UML?
› fairly widely (ab)used in industry
› some tool support

what’s bad about UML?
› tools are very weak
› semantics is complex and ill-defined