Ontology-Driven Conceptual Modeling

Giancarlo Guizzardi
Computer Science Department
Federal University of Espírito Santo (UFES), Brazil

IAOA Summer School
Trento, Italy
Day 3
Distinctions Among Categories of Object Types

- Type
  - ObjectType
    - Sortal Type
      - Rigid Sortal Type
        - Kind
          - {Person}
        - subKind
          - {Man, Woman}
      - Anti-Rigid Sortal Type
        - {Student, Teenager, FootballPlayer}
    - Non-Sortal Type
      - {Insurable Item}
Relational Dependence (D+)

- A type T is relationally dependent on another type P via relation R iff for every instance x of T there is an instance y of P such that x and y are related via R:

\[ D^+(T, P, R) = \text{def} \ \Box (\forall x \ T(x) \rightarrow \exists y \ P(y) \land R(x, y)) \]

\[ D^+(\text{Student, School, Enrolled-at}) = \text{def} \ \Box (\forall x \ \text{Student}(x) \rightarrow \exists y \ \text{School}(y) \land \text{Enrolled-at}(x, y)) \]
Distinctions Among Categories of Object Types

- **Type**
  - **ObjectType**
    - **Sortal Type**
      - **Rigid Sortal Type**
        - **Kind**
          - {Person}
        - **subKind**
          - {Man, Woman}
      - **Anti-Rigid Sortal Type**
        - **Phase**
          - {Teenager, LivingPerson}
        - **Role**
          - {Student, Husband}
  - **Non-Sortal Type**
    - {Insurable Item}
Roles

```
Enrolled-at
```

Student <-> School

*
The Relational property in this case is part of the very definition of the Role:

\[
\text{Student}(x) = \text{def} \ Person(x) \land \exists y \text{ School}(y) \\
\land \text{enrolled-at}(x,y)
\]
• Defined as a (anti-rigid) specialization of a kind such that the specialization condition is a relational one (correlated with derivation by participation)
Person

Student

School

enrolled-at

1..* 1..*

WORLD W’
«kind»Person

«role»Student

enrolled-at

1..* 1..*

«kind»School

WORLD W’’
Phases

• Defined as a (anti-rigid) specialization of a kind such that the specialization condition is an intrinsic one
• Phases are always defined in a so-called Phase Partition
• Phase Partitions are partitions in strong sense, i.e., they are disjoint and complete generalization sets
Person

\{disjoint, complete\}

\«phase\» LivingPerson

\«phase\» DeceasedPerson

LivingPerson

Deceased Person

PERSON

WORLD W
Person

\{disjoint,complete\}

\textbf{\textit{\textless phase\textgreater}}
LivingPerson

\textbf{\textit{\textless phase\textgreater}}
DeceasedPerson

\textit{PERSON}\ W’’
Imagine a game quite like football...

- Fault, Forward Kick, Shoot, Assistance, Blocked Shoot, Defended Shoot, Running Forward Kick, Penalty Kick, Pass, Card Punishment, Goal, Direct Free Kick
Football Game Action

- Fault
- Card Punishment
- Forward Kick
  - Direct Free Kick
  - Running Forward Kick
    - Penalty kick
  - Shoot
    - Goal
    - Blocked Shoot
    - Defended Shoot
  - Assistance
  - pass
«role»
Customer

«kind»
Person
An anti-rigid type cannot be a supertype of a Rigid Type
WORLD W

Instance of

Student

Person

x

Instance of
Since Student is anti-rigid, there must be a world $W'$ such that $x$ is not an instance of Student in $W'$. 
But since Person is rigid then $x$ must be an instance of Person in all worlds, including in $W'$.
But, given the semantics of supertying, we have that in all worlds (including $W'$), whoever is a Person is a Student.
We run into a logical contradiction!
An anti-rigid type cannot be a supertype of a Rigid Type
## Different Categories of Object Types

<table>
<thead>
<tr>
<th>Category of Type</th>
<th>Supply Identity</th>
<th>Inherits Identity</th>
<th>Rigidity</th>
<th>Dependence</th>
</tr>
</thead>
<tbody>
<tr>
<td>SORTAL</td>
<td>-</td>
<td>+</td>
<td></td>
<td></td>
</tr>
<tr>
<td>« kind »</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>-</td>
</tr>
<tr>
<td>« subkind »</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>-</td>
</tr>
<tr>
<td>« role »</td>
<td>-</td>
<td>+</td>
<td>-</td>
<td>+</td>
</tr>
<tr>
<td>« phase »</td>
<td>-</td>
<td>+</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>NON-SORTAL</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Distinctions Among Object Types

- **ObjectType**
  - **Sortal Type**
    - **Rigid Sortal Type**
      - **Kind**
      - **subkind**
    - **Anti-Rigid Sortal Type**
      - **Phase**
      - **Role**
  - **Non-Sortal Type**
    - **Rigid Non-Sortal Type**
    - **Anti-Rigid Non-Sortal Type**
Distinctions Among Object Types

```
Type

ObjectType

Sortal Type

Rigid Sortal Type

Kind
{Person}

subkind
{Brazilian, Man, Woman}

Anti-Rigid Sortal Type

Phase
{Student, Husband}

Role
{Teenager, LivingPerson}

Non-Sortal Type

Rigid Non-Sortal Type

Category
{Physical Object}

Anti-Rigid Non-Sortal Type
```
Categories (I-,R+)

- Categories are rigid non-sortals representing necessary (in the modal sense) features of entities of different kinds.
- Typically, they form the top-most layer of object types in an ontology.

```
[Diagram]
```

```
Categories (I-,R+)

- Categories are rigid non-sortals representing necessary (in the modal sense) features of entities of different kinds.
- Typically, they form the top-most layer of object types in an ontology.
```
Like all non-sortals, categories are always defined as abstract classes, which means that all categories must have at least a kind as a subtype.
Categories

- A kind can be subtype of multiple categories and a category can be a supertype of multiple kinds.
- Since all kinds are disjoint (otherwise the instances in their intersection would inherit multiple principles of identity), all subtypes of a category form a disjoint generalization set.
- On the other hand, since categories represent features of multiple kinds, they are very seldom complete.
Distinctions Among Object Types

- **Type**
  - **ObjectType**
    - **Sortal Type**
      - **Rigid Sortal Type**
        - **Kind**
          - {Person}
        - **subkind**
          - {Brazilian, Man, Woman}
      - **Anti-Rigid Sortal Type**
        - **Phase**
          - {Student, Husband}
        - **Role**
          - {Teenager, LivingPerson}
    - **Non-Sortal Type**
      - **Rigid Non-Sortal Type**
        - **Category**
          - {Physical Object}
      - **Anti-Rigid Non-Sortal Type**
        - **RoleMixin**
          - {Crime Weapon}
RoleMixin(I-, R-, D+)

- RoleMixen are anti-rigid and relational dependent non-sortals representing contingent (in the modal sense) features of entities of different kinds.
- As all Non-Sortals, RoleMixins classify entities belonging to different kinds.
- Like all non-sortals, categories are always defined as abstract classes.
Roles with Disjoint Allowed Types

«role» Customer

Person  Organization
Roles with Disjoint Allowed Types

Person

Organization

«role» Customer
Roles with Disjoint Admissible Types

«roleMixin»

Customer
Roles with Disjoint Allowed Types

```
«roleMixin»
Customer
«role»
PersonalCustomer
<role>
CorporateCustomer
```
Roles with Disjoint Allowed Types
Roles with Disjoint Admissible Types
Which one is better?

by Chris Welty
The Pattern in ORM

by Terry Halpin
Patterns and Metaphors

• From a modeling viewpoint, patterns can be seen as higher-granularity modeling primitives

• Design Patterns are the modeling counterpart of a device for Structure Transferability capturing a domain-independent system of types and relations that offer a standardized solution for a recurrent problem
RoleMixins

- A RoleMixin cannot be a super-type of a category, in fact, an Anti-Rigid Non-Sortal type cannot be a supertype of a rigid one.
RoleMixin

PERSON

CUSTOMER

PERSONAL CUSTOMER

CORPORATE CUSTOMER

ORGANIZATION

WORLD W

Supplier
RoleMixin

PERSON
- Personal Customer

CUSTOMER
- Corporate Customer

ORGANIZATION

SUPPLIER

WORLD W'
Distinctions Among Object Types

- **ObjectType**
- **Sortal Type**
  - **Rigid Sortal Type**
    - **Kind**
      - {Person}
    - **subkind**
      - {Brazilian, Man, Woman}
  - **Anti-Rigid Sortal Type**
  - **Rigid Non-Sortal Type**
    - **Phase**
      - {Student, Husband}
    - **Role**
      - {Teenager, LivingPerson}
    - **Category**
      - {Physical Object}
  - **Non-Rigid Non-Sortal Type**
    - **Anti-Rigid Non-Sortal Type**
    - **Semi-Rigid Non-Sortal Type**
      - **RoleMixin**
      - **Mixin**
        - {Insurable Item}
Modal Meta-Properties

- We will discuss here four types of Ontological Meta-Properties which are of a modal nature:

  - RIGIDITY
  - NON-RIGIDITY
  - ANTI-RIGIDITY
  - SEMI-RIGIDITY
Non-Rigidity

- Anti-Rigidity is not the logical negation of rigidity, non-rigidity is! Anti-Rigidity is stronger than that as a logical constraint

\[ R^+(T) = \text{def} \, \Box (\forall x \, T(x) \rightarrow \Box (T(x))) \]

\[ R^-(T) = \text{def} \, \Diamond (\exists x \, T(x) \land \Diamond \neg T(x)) \]
Non-Rigidity

- Anti-Rigidity is not the logical negation of rigidity, non-rigidity is! Anti-Rigidity is stronger than that as a logical constraint.

\[
R^+(T) = \text{def } \Box (\forall x \; T(x) \rightarrow \Box (T(x)))
\]

Logical Negation

\[
R^-(T) = \text{def } \Diamond (\exists x \; T(x) \land \Diamond \neg T(x))
\]

\[
R^\sim(T) = \text{def } \Box (\forall x \; T(x) \rightarrow \Diamond (\neg T(x)))
\]
Non-Rigidity

- Anti-Rigidity is not the logical negation of rigidity, non-rigidity is! Anti-Rigidity is stronger than that as a logical constraint

\[
\text{Rigidity} \quad R^+(T) =_{\text{def}} \Box(\forall x \ T(x) \rightarrow \Box(T(x)))
\]

\[
\downarrow \quad \text{Logical Negation}
\]

\[
\text{Non-Rigidity} \quad R^{-}(T) =_{\text{def}} \Diamond(\exists x T(x) \land \Diamond \neg T(x))
\]

\[
\text{Anti-Rigidity} \quad R^{-}(T) =_{\text{def}} \Box(\forall x \ T(x) \rightarrow \Diamond(\neg T(x)))
\]

\[
\text{Semi-Rigidity} \quad R^{\sim}(T) =_{\text{def}} R^{-}(T) \land \neg R^{\sim}(T)
\]
**Mixin(I-,R~)**

- Mixins are semi-rigid non-sortals, i.e., they represent features which are necessary (in the modal sense) for some of its instances but contingent to others

\[ SR(T) = \text{def} \ NR(T) \land \neg R-(T) \]

- Every Mixin must be a supertype of a rigid type as well as a supertype of a Non-Rigid type (typically a phase)
Mixins

- Mixins represent semi-rigid features of multiple kinds.
- Since all kinds are disjoint, all subtypes of a mixin form a disjoint generalization set. On the other, like in the case of categories, these generalization sets are very seldom complete.

```
{mixin} InsuredItem
    \{disjoint\}
    \triangleleft

{kind} Car

{kind} House
    \{disjoint,complete\}
    \triangleleft

{phase} InsuredHouse

{phase} NonInsuredHouse
```
Mixin

WORLD W

INSURABLE ITEM

HOUSE

CAR

Insured house

Uninsured house

Uninsured house
## Different Categories of Object Types

<table>
<thead>
<tr>
<th>Category of Type</th>
<th>Supply</th>
<th>Identity</th>
<th>Rigidity</th>
<th>Dependence</th>
</tr>
</thead>
<tbody>
<tr>
<td>SORTAL</td>
<td>-</td>
<td>+</td>
<td></td>
<td></td>
</tr>
<tr>
<td>« kind »</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>-</td>
</tr>
<tr>
<td>« subkind »</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>-</td>
</tr>
<tr>
<td>« role »</td>
<td>-</td>
<td>+</td>
<td>-</td>
<td>+</td>
</tr>
<tr>
<td>« phase »</td>
<td>-</td>
<td>+</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>NON-SORTAL</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>« category »</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>-</td>
</tr>
<tr>
<td>« roleMixin »</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>+</td>
</tr>
<tr>
<td>« mixin »</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Stereotype</td>
<td>Constraints</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>----------------</td>
<td>--------------------------------------------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>RIGID SORTALS</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
| « kind »       | supertype is **not** a member of {« subkind », « phase », « role », « roleMixin »}  
|                | For every subkind SK there is a unique kind K such that K is a supertype of SK |
| « subkind »    | supertype is **not** a member of {« phase », « role », « roleMixin »}  
|                | For every subkind SK there is a unique kind K such that K is a supertype of SK |
| **ANTI-RIGID SORTALS** |                                             |
| « phase »      | Always defined as part of disjoint and complete partition.  
|                | For every Phase P there is a unique Kind K such that K is a supertype of P |
| « role »       | Let X be a class stereotyped as « role » and R be an association representing X's restriction condition. Then, #X.R \( \geq 1 \)  
|                | For every Role X there is a unique Kind K such that K is a supertype of X |
| **NON-SORTALS** |                                               |
| « category »   | supertype is **not** a member of {« kind », « subkind », « phase », « role », « roleMixin »}  
|                | Always defined as an abstract class  
|                | Always specialized by a unique kind  
|                | Always defined in a disjoint partition |
| « roleMixin »  | supertype is **not** a member of {« kind », « subkind », « phase », « role »}.  
|                | Let X be a class stereotyped as « roleMixin » and R be an association representing X’s restriction condition. Then, #X.R \( \geq 1 \)  
|                | Always defined as an abstract class  
|                | Always defined in a disjoint partition  
|                | Always specialized by Sortals |
| « mixin »      | supertype is **not** a member of {« kind », « subkind », « phase », « role », « roleMixin »}  
|                | Always defined as an abstract class |
The requested action violates the integrity of the model.
Reason: An anti-rigid instance cannot be a supertype of a rigid instance.
ObjectClass instances which are not abstract (isAbstract = false) and which are not instances of a Kind must have a Kind as supertype.

Role instances must be connected to at least one individual via its characterizing relation.
Ontology-Derived Patterns

```
«roleMixin»
A
«role»
B

Obj.Type F
Sortal D
Sortal E

«role»
C
1..*
1..*

min ≥ 1

S

{disjoint, complete}

«phase»X
...
«phase»Pn

«role»X

R
min..max
T
```
PARTHOOD
Parthood

• Part-whole relations are fundamental from a cognitive perspective, i.e., for the realization of many important cognitive tasks.

• Moreover, and also for this reason, parthood is a relation of significant importance in conceptual modeling, being present in practically all conceptual/object-oriented modeling languages (e.g., OML, UML, EER)

• Although it has not yet been adopted as a modeling primitive in the semantic web languages, there is a significant body of work discussing its relevance for reasoning in description logics

• From now on, we use the symbol (<) to represent the part-whole relation
Parthood is irreflexive, i.e., nothing is part of itself
\[ \forall x \neg(x < x) \]
Ground Mereology

Parthood is irreflexive, i.e., nothing is part of itself
\[ \forall x \neg(x < x) \]

Parthood is anti-symmetric, i.e., if X is part of Y then Y cannot be part of X
\[ \forall x, y (x < y) \rightarrow \neg(y < x) \]
Ground Mereology

Parthood is irreflexive, i.e., nothing is part of itself
\[ \forall x \neg (x < x) \]

Parthood is anti-symmetric, i.e., if X is part of Y then Y cannot be part of X
\[ \forall x, y (x < y) \rightarrow \neg (y < x) \]

Parthood is transitive, i.e., if X is part of Y and Y is part of Z then X is part of Z
\[ \forall x, y, z (x < y) \land (y < z) \rightarrow (x < z) \]
Minimum Mereology

- The formal semantics just present defines a so-called *strict partial order* relation
- These axioms are not sufficient to differentiate parthood from other partial order relations (e.g., less-than, bigger-than, causality, strict temporal precedence)
- A stronger theory named Minimum Mereology, thus, defines some additional notions
Two entities overlap if they share a part

\[(x \cdot y) =_{\text{def}} \exists z \ (z \leq x) \land (z \leq y)\]

Notice that this includes the case in which one is part of the other.
Disjointness

Two entities are disjoint if they do not overlap

\[(x \cap y) =_{\text{def}} \neg(x \cdot y)\]
Weak Supplementation

- Minimum Mereology takes the partial order axioms of Ground Mereology and includes the so-called WSP (Weal Supplementation Property)
Weak Supplementation

Weak Supplementation Principle

If Y is part of X and if parthood is irreflexive
Then there must be another part of X which
is complementary to Y

∀x,y (y < x) → ∃z (z < x) ∧ (z \dashv y)
Weak Supplementation Principle
Weak Supplementation Principle

Diagram:

1. Event
2. Event

Symbols:

- *: Multiplicity indicator
- 1..*: Multiplicity range

Note: The diagrams illustrate the principle with red crosses indicating invalid configurations.
Weak Supplementation Pattern

```
Entity
{disjoint, complete}

AtomicEntity  ComplexEntity

2..*

*  *
```
Weak Supplementation Pattern

AtomicEntity\( (x) =_{\text{def}} \neg \exists y \ (y < x) \)
One of the representations of parthood in UML termed **Aggregation**

- The hollow diamond is connected to the whole
- Aggregation is supposed to be a irreflexive and anti-symmetric relation
From a formal perspective, the most used theory of parthood is the Extensional Mereology, which can be obtained from the Minimum Mereology by including a Strong Supplementation Principle.

Extensional Mereology (+Strong Supplementation Principle)

Minimum Mereology (+WSP)

Ground Mereology (Irreflexivity, Anti-Symmetry, Transitivity)
Extensional Mereology

• The Strong Supplementation Principle implies something called the **Extensional Principle** which implies that: two entities are identical if they have the same parts

\[
(((x=y) \leftrightarrow \forall z ((z < x) \leftrightarrow (z < y))))
\]
Problems from a Conceptual Modeling point of view

• Ground Mereology
  – Unrestricted transitivity of parthood
Claudio's Brain

Claudio

LOA
Problems from a CM point of view

- **Extensional Mereology**
  - Extensional Principle of Identity which makes all parts of an entity as *essential parts*
  - In other words, every object is defined by the sum of its. Thus, we have that:
    - (i) The change of any of the parts changes the identity of the object
Problems from a CM point of view

• Extensional Mereology
  – Extensional Principle of Identity which makes all parts of an entity as *essential parts*
  – In other words, every object is defined by the sum of its. Thus, we have that:
    • (ii) Two objects are the same if and only if they have the same parts

The Beatles = The Liverpool Indoors Football Team
Problems from a CM point of view

• Extensional Mereology
  – Failure to take into account the different roles that parts play within the whole. In other words, not all parts have the same importance with respect to the whole: some parts are optional, some parts can be replaced by others of the same kind, some parts cannot be replaced without causing the entity to change its class. Finally, some parts cannot be replaced at all, i.e., without altering the identity of the whole
Problems from a CM point of view

• **Extensional Mereology**
  
  – There are other meta-properties that can be used to qualify the relation between parts and wholes. An common one is the distinction between shared or non-shared parts.
Problems from a CM point of view

• **Extensional Mereology**
  – There are other meta-properties that can be used to qualify the relation between parts and wholes. A common one is the distinction between shared or non-shared parts.

![Diagram of Composition in UML]

• The other representations of parthood in UML is termed **Composition**
• The black diamond is connected to the whole
• Composition is supposed to be an irreflexive, anti-symmetric and transitive relation
• It is also supposed to imply non-shareability of parts and lifetime dependence from the part to the whole
• As we will see, the notions of non-shareability, lifetime dependence and transitivity are orthogonal!
Problems from a Conceptual Modeling point of view

• In general, mereological theories treat aggregates (wholes) roughly as *sets of entities*

• As a consequence, the conditions that bind all the parts of a whole together are minimal. Thus, a whole formed by any arbitrary sum of entities is considered as good as any other (e.g., the aggregate of number 3, van Gogh’s ear and the third act of Turandot)

• However, humans only accept the aggregation of entities if the resulting aggregate plays some role in their conceptual schemes, i.e., if the wholes they form represent genuine categories

• We need then genuine unifying (characterizing) relations for *Integral Wholes*
Integral Wholes

• We must complement mereology (the theory of parts) with a *theory of wholes*, in which the relations that tie the parts of a whole together are also considered.
Not one but several parthood relations

• Studies in Cognitive Science and Linguistics have shown that we do not have a single notion of part, but multiple parthood relations forming different types of aggregates
  – (a) Quantities
    • Subquantity of (alcohol-wine)
  – (b) Collectives
    • Member of (a specific tree – the black forest)
    • Subcollective of (the north part of the black forest – the black forest)
  – (c) Functional Entities
    • Component of (heart-circulatory system)
Not one but several parthood relations

• These different notions of part are neither orthogonal to the previously mentioned mereological axioms nor to the different meta-properties that can be applied to part-whole relations. Moreover, different notions of parts are related to different sorts of unifying relations.
gguizzardi@inf.ufes.br