# COMP718: Ontologies and Knowledge Bases

## Lecture 6: Top-down Ontology Development II

Maria Keet

email: keet@ukzn.ac.za

home: http://www.meteck.org

School of Mathematics, Statistics, and Computer Science
University of KwaZulu-Natal, South Africa

6/13 March 2012

# Outline

# Outline

1. Parts, mereology, meronymy
   - Introduction
   - Mereology
   - Implementation
   - Meronymy
2. Taxonomy of types of part-whole relations
   - The taxonomy
   - Using the taxonomy of part-whole relations
   - RBox Compatibility
3. Extensions
4. Ontology Design Patterns

# Some questions and problems (not exhaustive...)

- Is Tshwane a more specific instance of Gauteng, or a part of it?
- Is a tunnel part of the mountain?
- What is the difference, if any, between how `Cell nucleus` and `Cell` are related and how `Receptor` and `Cell wall` are related?
- And w.r.t. `Brain part of Human` and/versus `Hand part of Boxer`? (assuming boxers must have their own hands)
- A classical example: hand is part of musician, musician part of orchestra, but clearly, the musician's hands are not part of the orchestra. Is part-of then not transitive, or is there a problem with the example?

# Some questions and problems (not exhaustive...)

- Is Tshwane a more specific instance of Gauteng, or a part of it?
- Is a tunnel part of the mountain?
- What is the difference, if any, between how `Cell nucleus` and `Cell` are related and how `Receptor` and `Cell wall` are related?
- And w.r.t. `Brain part of Human` and/versus `Hand part of Boxer`? (assuming boxers must have their own hands)
- A classical example: hand is part of musician, musician part of orchestra, but clearly, the musician's hands are not part of the orchestra. Is part-of then not transitive, or is there a problem with the example?

# Some questions and problems (not exhaustive...)

- Is Tshwane a more specific instance of Gauteng, or a part of it?
- Is a tunnel part of the mountain?
- What is the difference, if any, between how `Cell nucleus` and `Cell` are related and how `Receptor` and `Cell wall` are related?
- And w.r.t. `Brain part of Human` and/versus `Hand part of Boxer`? (assuming boxers must have their own hands)
- A classical example: hand is part of musician, musician part of orchestra, but clearly, the musician's hands are not part of the orchestra. Is part-of then not transitive, or is there a problem with the example?

# Some questions and problems (not exhaustive...)

- Is Tshwane a more specific instance of Gauteng, or a part of it?
- Is a tunnel part of the mountain?
- What is the difference, if any, between how `Cell nucleus` and `Cell` are related and how `Receptor` and `Cell wall` are related?
- And w.r.t. `Brain part of Human` and/versus `Hand part of Boxer`? (assuming boxers must have their own hands)
- A classical example: hand is part of musician, musician part of orchestra, but clearly, the musician's hands are not part of the orchestra. Is part-of then not transitive, or is there a problem with the example?

# Some questions and problems (not exhaustive...)

- Is Tshwane a more specific instance of Gauteng, or a part of it?
- Is a tunnel part of the mountain?
- What is the difference, if any, between how `Cell nucleus` and `Cell` are related and how `Receptor` and `Cell wall` are related?
- And w.r.t. `Brain part of Human` and/versus `Hand part of Boxer`? (assuming boxers must have their own hands)
- A classical example: hand is part of musician, musician part of orchestra, but clearly, the musician's hands are not part of the orchestra. Is part-of then not transitive, or is there a problem with the example?

# Analysis of the issues from diverse angles

- Mereological theories (Varzi, 2004), usage & extensions (e.g. mereotopology, relation with granularity, set theory)
- Early attempts with direct parthood, SEP triples, and other outstanding issues, some still remaining
- Cognitive & linguistic issues from meronymy
- Usage in conceptual modelling and ontology engineering
- Subject domains: thus far, mainly geo, bio, medicine

# Analysis of the issues from diverse angles

- Mereological theories (Varzi, 2004), usage & extensions (e.g. mereotopology, relation with granularity, set theory)
- Early attempts with direct parthood, SEP triples, and other outstanding issues, some still remaining
- Cognitive & linguistic issues from meronymy
- Usage in conceptual modelling and ontology engineering
- Subject domains: thus far, mainly geo, bio, medicine

# Analysis of the issues from diverse angles

- Mereological theories (Varzi, 2004), usage & extensions (e.g. mereotopology, relation with granularity, set theory)
- Early attempts with direct parthood, SEP triples, and other outstanding issues, some still remaining
- Cognitive & linguistic issues from meronymy
- Usage in conceptual modelling and ontology engineering
- Subject domains: thus far, mainly geo, bio, medicine

# Ground Mereology

Reflexivity (everything is part of itself)

$$\forall x(part\_of(x,x)) \quad (1)$$

Antisymmetry (two distinct things cannot be part of each other, or: if they are, then they are the same thing)

$$\forall x,y((part\_of(x,y) \land part\_of(y,x)) \rightarrow x = y) \quad (2)$$

Transitivity (if x is part of y and y is part of z, then x is part of z)

$$\forall x,y,z((part\_of(x,y) \land part\_of(y,z)) \rightarrow part\_of(x,z)) \quad (3)$$

Proper parthood

$$\forall x,y(proper\_part\_of(x,y) \equiv part\_of(x,y) \land \neg part\_of(y,x)) \quad (4)$$

# Ground Mereology

Proper parthood

$$\forall x, y(proper\_part\_of(x, y) \equiv part\_of(x, y) \wedge \neg part\_of(y, x)) \quad (5)$$

Asymmetry (if $x$ is part of $y$ then $y$ is not part of $x$)

$$\forall x, y(part\_of(x, y) \rightarrow \neg part\_of(y, x)) \quad (6)$$

Irreflexivity ($x$ is not part of itself)

$$\forall x \neg(part\_of(x, x)) \quad (7)$$

# Defining other relations with *part_of*

Overlap (x and y share a piece z)

$$\forall x, y(overlap(x, y) \equiv \exists z(part\_of(z, x) \wedge part\_of(z, y))) \tag{8}$$

Underlap (x and y are both part of some z)

$$\forall x, y(underlap(x, y) \equiv \exists z(part\_of(x, z) \wedge part\_of(y, z))) \tag{9}$$

Over- & undercross (over/underlap but not part of)

$$\forall x, y(overcross(x, y) \equiv overlap(x, y) \wedge \neg part\_of(x, y)) \tag{10}$$

$$\forall x, y(undercross(x, y) \equiv underlap(x, y) \wedge \neg part\_of(y, x)) \tag{11}$$

Proper overlap & Proper underlap

$$\forall x, y(p\_overlap(x, y) \equiv overcross(x, y) \wedge overcross(y, x)) \tag{12}$$

$$\forall x, y(p\_underlap(x, y) \equiv undercross(x, y) \wedge undercross(y, x)) \tag{13}$$

- With x as part, what to do with the remainder that makes up y?
    - Weak supplementation: every proper part must be supplemented by another, disjoint, part. **MM**
    - Strong supplementation: if an object fails to include another among its parts, then there must be a remainder. **EM**
- Problem with EM: non-atomic objects with the same proper parts are identical, because of this (extensionality principle), but sameness of parts may not be sufficient for identity E.g.: two objects can be distinct purely based on arrangement of its parts, differences statue and its marble (multiplicative approach)

- With x as part, what to do with the remainder that makes up y?
  - Weak supplementation: every proper part must be supplemented by another, disjoint, part. **MM**
  - Strong supplementation: if an object fails to include another among its parts, then there must be a remainder. **EM**
- Problem with EM: non-atomic objects with the same proper parts are identical, because of this (extensionality principle), but sameness of parts may not be sufficient for identity E.g.: two objects can be distinct purely based on arrangement of its parts, differences statue and its marble (multiplicative approach)

# General Extensional Mereology

- Strong supplementation [EM]

$$\neg part\_of(y,x) \rightarrow \exists z(part\_of(z,y) \wedge \neg overlap(z,x)) \quad (14)$$

- And add unrestricted fusion [GEM]. Let $\phi$ be a property or condition, then for every satisfied $\phi$ there is an entity consisting of all entities that satisfy $\phi$. [1] Then:

$$\exists x\phi \rightarrow \exists z \forall y(overlap(y,z) \leftrightarrow \exists x(\phi \wedge overlap(y,x))) \quad (15)$$

- Note that in EM and upward we have identity, from which one can prove acyclicity for ppo
- There are more mereological theories, and the above is not uncontested (more about that later)

[1] Need to refer to classes, but desire to stay within FOL. Solution: axiom schema with only predicates or open formulas

# Relations between common mereological theories

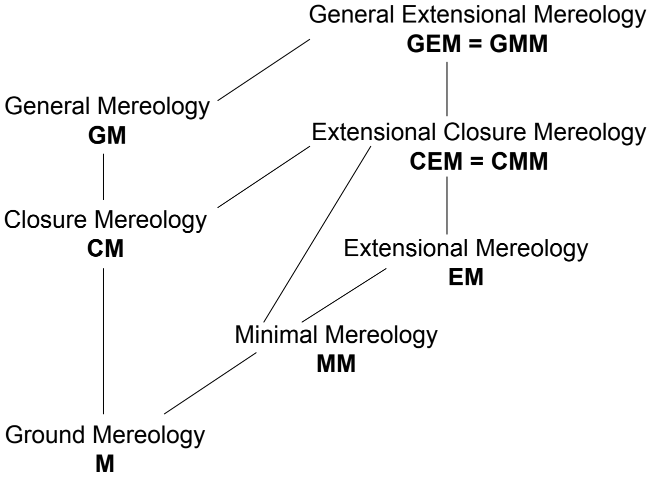



**Fig. 1:** Hasse diagram of mereological theories; from weaker to stronger, going uphill (after [44]).

*Can any of this be represented in a decidable fragment of first order logic for use in ontologies and (scalable) software implementations?*

# Things are improving...

- Early days (90s) and simplest options: DL-role `R` as `partof`, or `has-part` added as primitive role as $\succeq$, model it as the transitive closure of a parthood relation (16) and define e.g. `Car` as having wheels that in turn have tires (17):

$$\succeq \doteq (\texttt{primitive-part})* \tag{16}$$

$$\texttt{Car} \doteq \exists \succeq .(\texttt{Wheel} \sqcap \exists \succeq .\texttt{Tire}) \tag{17}$$

  Then $\texttt{Car} \sqsubseteq \exists \succeq .\texttt{Tire}$
- SEP triples with $\mathcal{ALC}$
- What $\mathcal{SHIQ}$ fixes cf. $\mathcal{ALC}$: Transitive roles, Inverse roles (to have both part-of and has-part), Role hierarchies (e.g. for subtypes of part-of), qualified Number restrictions (e.g. to represent that a bycicle has-part 2 wheels)
- Build-your-own DL-language

# What we can(not) implement now with DL-based ontology languages

Table: Properties of parthood and proper parthood compared to their support in $\mathcal{DLR}_\mu$, $\mathcal{SHOIN}$ and $\mathcal{SROIQ}$. $^*$: properties of the parthood relation (in M); $^\ddagger$: properties of the proper parthood relation (in M).

| **Language** $\Rightarrow$ <br> **Feature** $\Downarrow$ | $\mathcal{DLR}_\mu$ | $\mathcal{SHOIN}$ <br> ($\sim$ OWL-DL) | $\mathcal{SROIQ}$ <br> ($\sim$ OWL 2 DL) | DL-Lite$_A$ <br> ($\sim$ OWL 2 QL) |
|---|---|---|---|---|
| Reflexivity $^*$ | + | − | + | − |
| Antisymmetry $^*$ | − | − | − | − |
| Transitivity $^*$ $^\ddagger$ | + | + | + | − |
| Asymmetry $^\ddagger$ | + | + | + | + |
| Irreflexivity $^\ddagger$ | + | − | + | − |
| *Acyclicity* | + | − | − | − |

# Definitions in OBO Relations Ontology

- Instance-level relations
  - $c$ **part_of** $c_1$ at $t$ - a primitive relation between two continuant instances and a time at which the one is part of the other
  - $p$ **part_of** $p_1$, $r$ **part_of** $r_1$ - a primitive relation of parthood, holding independently of time, either between process instances (one a subprocess of the other), or between spatial regions (one a subregion of the other)
  - $c$ **contained_in** $c_1$ at $t \triangleq c$ **located_in** $c_1$ at $t$ and not $c$ **overlap** $c_1$ at $t$
  - c located_in r at t - a primitive relation between a continuant instance, a spatial region which it occupies, and a time

# Definitions in OBO Relations Ontology

- Class-level relations
  - $C$ *part_of* $C_1 \triangleq$ for all $c$, $t$, if $Cct$ then there is some $c_1$ such that $C_1 c_1 t$ and $c$ **part_of** $c_1$ **at** $t$.
  - $P$ *part_of* $P_1 \triangleq$ for all $p$, if $Pp$ then there is some $p_1$ such that: $P_1 p_1$ and $p$ **part_of** $p_1$.
  - $C$ *contained_in* $C_1 \triangleq$ for all $c$, $t$, if $Cct$ then there is some $c_1$ such that: $C_1 c_1 t$ and $c$ **contained_in** $c_1$ at $t$
- Need to commit to a foundational ontology. Recently, linked to BFO http://obofoundry.org/ro/#mappings (test release)
- Same labels, different relata and only a textual constraint: Label the relations differently

# Definitions in OBO Relations Ontology

- Class-level relations
  - $C$ *part_of* $C_1 \triangleq$ for all $c$, $t$, if $Cct$ then there is some $c_1$ such that $C_1c_1t$ and $c$ **part_of** $c_1$ **at** $t$.
  - $P$ *part_of* $P_1 \triangleq$ for all $p$, if $Pp$ then there is some $p_1$ such that: $P_1p_1$ and $p$ **part_of** $p_1$.
  - $C$ *contained_in* $C_1 \triangleq$ for all $c$, $t$, if $Cct$ then there is some $c_1$ such that: $C_1c_1t$ and $c$ **contained_in** $c_1$ at $t$
- Need to commit to a foundational ontology. Recently, linked to BFO http://obofoundry.org/ro/#mappings (test release)
- Same labels, different relata and only a textual constraint: Label the relations differently

# Linguistic use of part-whole relations

- Part of?
  - ⋆ Centimeter part of Decimeter
  - ⋆ Decimeter part of Meter
  - — *therefore* Centimeter part of Meter
  - ⋆ Meter part of SI
  - — but *not* Centimeter part of SI
- Transitivity?
  - ⋆ Person part of Organisation
  - ⋆ Organisation located in Bolzano
  - — therefore Person located in Bolzano?
  - — but *not* Person part of Bolzano

# Linguistic use of part-whole relations

- Part of?
  - ⋆ Centimeter part of Decimeter
  - ⋆ Decimeter part of Meter
  - — *therefore* Centimeter part of Meter
  - ⋆ Meter part of SI
  - — but *not* Centimeter part of SI
- Transitivity?
  - ⋆ Person part of Organisation
  - ⋆ Organisation located in Bolzano
  - — therefore Person located in Bolzano?
  - — but *not* Person part of Bolzano

# Linguistic use of part-whole relations (meronymy)

- Part of?
  - ⋆ Centimeter part of Decimeter
  - ⋆ Decimeter part of Meter
  - — *therefore* Centimeter part of Meter
  - ⋆ Meter part of SI
  - — but *not* Centimeter part of SI
- Transitivity?
  - ⋆ Person member of Organisation
  - ⋆ Organisation located in Bolzano
  - — therefore Person located in Bolzano?
  - — but *not* Person member of Bolzano

# Linguistic use of part-whole relations

- Which part of?
  - ⋆ CellMembrane structural part of RedBloodCell
  - ⋆ RedBloodCell part of Blood
  - — but *not* CellMembrane structural part of Blood
  - ⋆ Receptor structural part of CellMembrane
  - — *therefore* Receptor structural part of RedBloodCell

# Linguistic use of part-whole relations

- Which part of?
  - ⋆ CellMembrane structural part of RedBloodCell
  - ⋆ RedBloodCell contained in? Blood
  - — but *not* CellMembrane structural part of Blood
  - ⋆ Receptor structural part of CellMembrane
  - — *therefore* Receptor structural part of RedBloodCell

# Addressing the issues

- Efforts to disambiguate this confusion; e.g. an informal taxonomy by Winston et al (1987), list of 6 types motivated by UML conceptual modeling (Odell) ontology-inspired conceptual modelling (Guizzardi)
- Location, containment, membership of a collective, quantities of a mass
- Relatively well-settled debate on transitivity, or not

# Outline

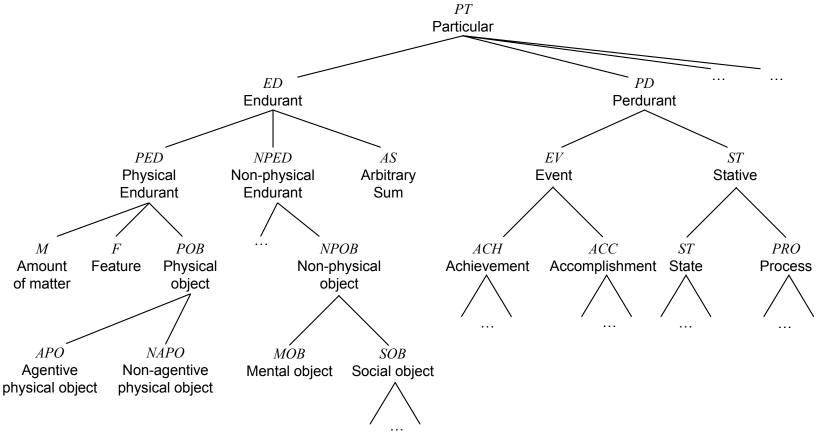
# Overview

- Mereological *part_of* (and subtypes) versus 'other' part-whole relations
- Categories of object types of the part-whole relation changes
- Structure these relations by (non/in)transitivity and kinds of relata
- Simplest mereological theory, **M**.
- Commit to a foundational ontology: DOLCE (though one also could choose, a.o., BFO, OCHRE, GFO, ...)

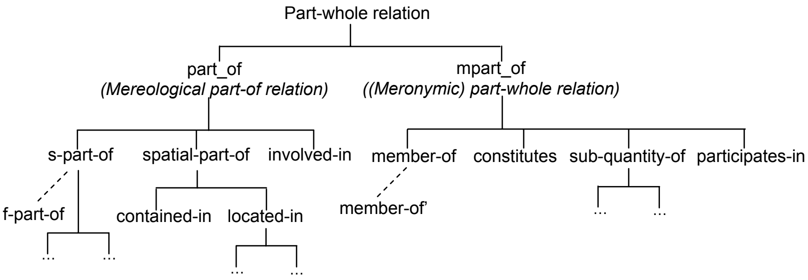
# Overview

- Mereological *part_of* (and subtypes) versus 'other' part-whole relations
- Categories of object types of the part-whole relation changes
- Structure these relations by (non/in)transitivity and kinds of relata
- Simplest mereological theory, **M**.
- Commit to a foundational ontology: DOLCE (though one also could choose, a.o., BFO, OCHRE, GFO, ...)

# DOLCE categories



# Part-whole relations



# Part-whole relations

"member-bunch", collective nouns (e.g. `Herd`, `Orchestra`) with their members (`Sheep`, `Musician`)

$$\forall x, y(member\_of_n(x, y) \triangleq mpart\_of(x, y) \wedge (POB(x) \vee SOB(x)) \wedge SOB(y))$$

"material-object", that what something is made of (e.g., `Vase` and `Clay`)

$$\forall x, y(constitutes_{it}(x, y) \equiv constituted\_of_{it}(y, x) \triangleq mpart\_of(x, y) \wedge POB(y) \wedge M(x))$$

# Part-whole relations

"quantity-mass", "portion-object", relating a smaller (or sub) part of an amount of matter to the whole. Two issues (glass of wine & bottle of wine vs. `Salt` as subquantity of `SeaWater`)

$$\forall x, y(sub\_quantity\_of_n(x, y) \triangleq mpart\_of(x, y) \wedge M(x) \wedge M(y))$$

"noun-feature/activity", entity participates in a process, like `Enzyme` that participates in `CatalyticReaction`

$$\forall x, y(participates\_in_{it}(x, y) \triangleq mpart\_of(x, y) \wedge ED(x) \wedge PD(y))$$

# Part-whole relations

processes and sub-processes (e.g. `Chewing` is involved in the grander process of `Eating`)

$$\forall x, y(involved\_in(x, y) \triangleq part\_of(x, y) \wedge PD(x) \wedge PD(y))$$

Object and its 2D or 3D region, such as `contained_in(John's address book, John's bag)` and `located_in(Pretoria, South Africa)`

$$\forall x, y(contained\_in(x, y) \triangleq part\_of(x, y) \wedge R(x) \wedge R(y) \wedge \\ \exists z, w(has\_3D(z, x) \wedge has\_3D(w, y) \wedge ED(z) \wedge ED(w)))$$

$$\forall x, y(located\_in(x, y) \triangleq part\_of(x, y) \wedge R(x) \wedge R(y) \wedge \\ \exists z, w(has\_2D(z, x) \wedge has\_2D(w, y) \wedge ED(z) \wedge ED(w)))$$

$$\forall x, y(s\_part\_of(x, y) \triangleq part\_of(x, y) \wedge ED(x) \wedge ED(y))$$

# Using the taxonomy of part-whole relations

- Representing it correctly in ontologies and conceptual data models
  - Decision diagram
  - Using the categories of the foundational ontology
  - Examples
  - *Software application* that simplifies all that
- Reasoning with a taxonomy of relations
  - The *RBox reasoning service* to pinpoint errors

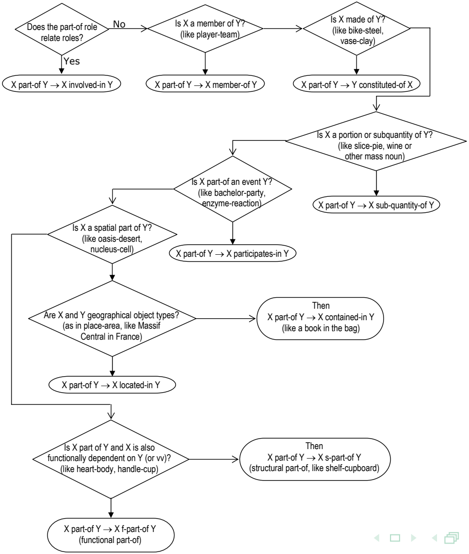
# Using the taxonomy of part-whole relations

- Representing it correctly in ontologies and conceptual data models
  - Decision diagram
  - Using the categories of the foundational ontology
  - Examples
  - *Software application* that simplifies all that
- Reasoning with a taxonomy of relations
  - The *RBox reasoning service* to pinpoint errors

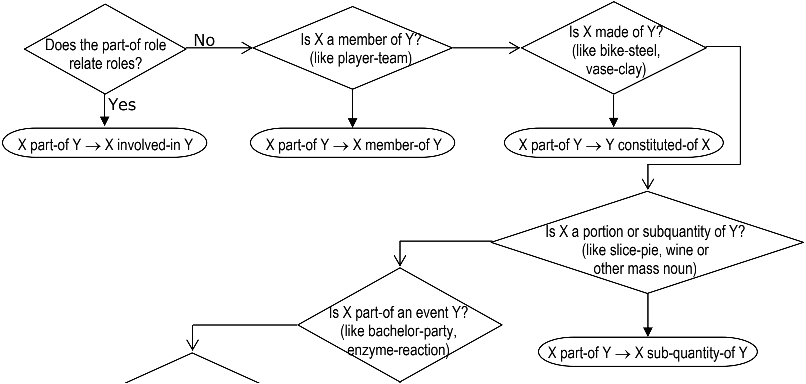
# Using the taxonomy of part-whole relations

- Representing it correctly in ontologies and conceptual data models
  - Decision diagram
  - Using the categories of the foundational ontology
  - Examples
  - *Software application* that simplifies all that
- Reasoning with a taxonomy of relations
  - The *RBox reasoning service* to pinpoint errors

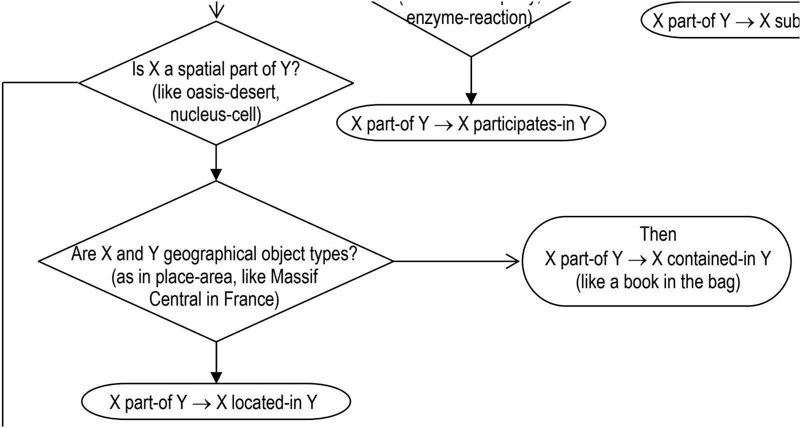
# Decision diagram



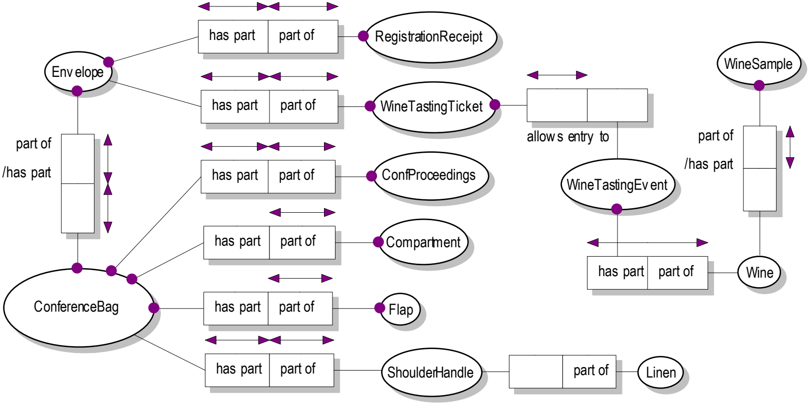
# Decision diagram



# Decision diagram



# Example - before



# Example - after

- Envelope is not involved-in, not a member-of, does not constitute, is not a sub-quantity of, does not participate-in, is not a geographical object, but instead is contained-in the ConferenceBag.
- Transitivity holds for the mereological relations: derived facts are automatically correct, like RegistrationReceipt contained-in ConferenceBag.
- Intransitivity of Linen and ConferenceBag, because a conference bag is not wholly constituted of linen (the model does not say what the Flap is made of).
- Completeness, i.e. that *all* parts make up the whole, is implied thanks to the closed-world assumption. ConferenceBag directly contains the ConfProceedings and Envelope *only*, and does not contain, say, the Flap.

# Example - after

- Envelope is not involved-in, not a member-of, does not constitute, is not a sub-quantity of, does not participate-in, is not a geographical object, but instead is contained-in the ConferenceBag.
- Transitivity holds for the mereological relations: derived facts are automatically correct, like RegistrationReceipt contained-in ConferenceBag.
- Intransitivity of Linen and ConferenceBag, because a conference bag is not wholly constituted of linen (the model does not say what the Flap is made of).
- Completeness, i.e. that *all* parts make up the whole, is implied thanks to the closed-world assumption. ConferenceBag directly contains the ConfProceedings and Envelope *only*, and does not contain, say, the Flap.

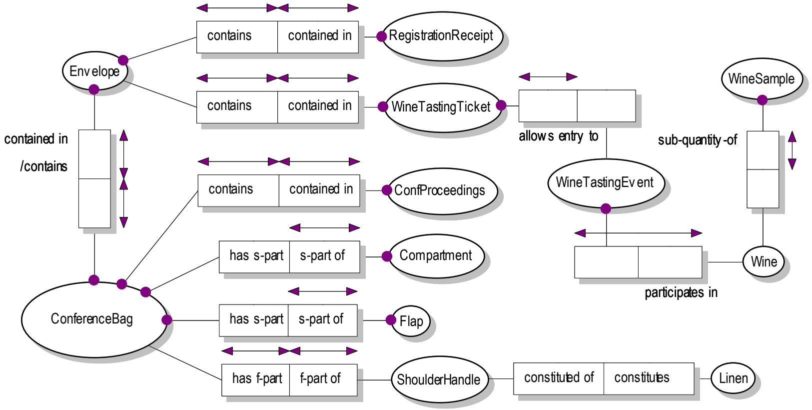
# Example - after

- Envelope is not involved-in, not a member-of, does not constitute, is not a sub-quantity of, does not participate-in, is not a geographical object, but instead is contained-in the ConferenceBag.
- Transitivity holds for the mereological relations: derived facts are automatically correct, like RegistrationReceipt contained-in ConferenceBag.
- Intransitivity of Linen and ConferenceBag, because a conference bag is not wholly constituted of linen (the model does not say what the Flap is made of).
- Completeness, i.e. that *all* parts make up the whole, is implied thanks to the closed-world assumption. ConferenceBag directly contains the ConfProceedings and Envelope *only*, and does not contain, say, the Flap.

# Example - after

- Envelope is not involved-in, not a member-of, does not constitute, is not a sub-quantity of, does not participate-in, is not a geographical object, but instead is contained-in the ConferenceBag.
- Transitivity holds for the mereological relations: derived facts are automatically correct, like RegistrationReceipt contained-in ConferenceBag.
- Intransitivity of Linen and ConferenceBag, because a conference bag is not wholly constituted of linen (the model does not say what the Flap is made of).
- Completeness, i.e. that *all* parts make up the whole, is implied thanks to the closed-world assumption. ConferenceBag directly contains the ConfProceedings and Envelope *only*, and does not contain, say, the Flap.

# Example - after



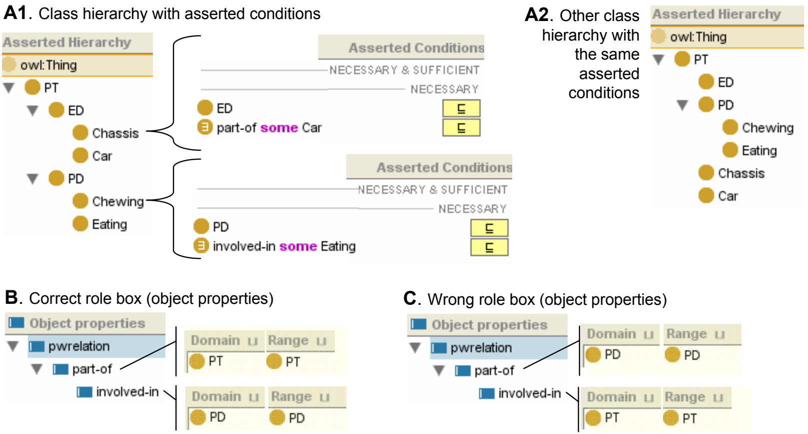
# Using DOLCE's categories

- The participating objects instantiate some category (*ED*, *PD*, etc)
- Given the formalization, one immediately can exclude/identify appropriate relations, taking a shortcut in the decision diagram
  - E.g.: *Chewing* and *Eating* are both a kind of (a subtype of) *PD*, hence *involved_in*
  - E.g.: *Alcohol* and *Wine* are both mass nouns, or *M*, hence *sub_quantity_of*
- Demo of OntoPartS

# Requirements for reasoning over the hierarchy

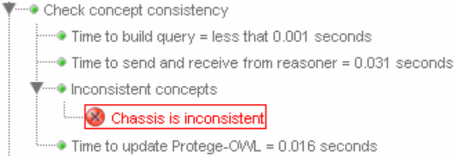
- Represent at least Ground Mereology,
- Express ontological categories and their taxonomic relations,
- Having the option to represent transitive and intransitive relations, and
- Specify the domain and range restrictions (/relata/entity types) for the classes participating in a relation.

# Current behaviour of reasoners

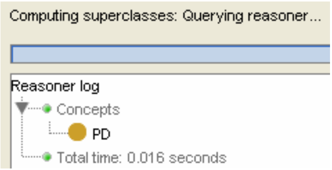


# Current behaviour of reasoners

1. A1+B+racer: *ontology OK*
2. A2+B+racer: *ontology OK*
3. A1+C+racer: class hierarchy is inconsistent




4. A2+C+racer: Chassis reclassified as PD



# The *RBox Compatibility* service – definitions

## Definition (Domain and Range Concepts)

Let $R$ be a role and $R \sqsubseteq C_1 \times C_2$ its associated Domain & Range axiom. Then, with the symbol $D_R$ we indicate the *User-defined Domain* of $R$—i.e., $D_R = C_1$—while with the symbol $R_R$ we indicate the *User-defined Range* of $R$—i.e., $R_R = C_2$.

## Definition (RBox Compatibility)

For each pair of roles, $R, S$, such that $\langle \mathcal{T}, \mathcal{R} \rangle \models R \sqsubseteq S$, check:

Test 1. $\langle \mathcal{T}, \mathcal{R} \rangle \models D_R \sqsubseteq D_S$ and $\langle \mathcal{T}, \mathcal{R} \rangle \models R_R \sqsubseteq R_S$;

Test 2. $\langle \mathcal{T}, \mathcal{R} \rangle \not\models D_S \sqsubseteq D_R$;

Test 3. $\langle \mathcal{T}, \mathcal{R} \rangle \not\models R_S \sqsubseteq R_R$.

An RBox is said to be compatible iff *Test 1* and (*2* or *3*) hold for all pairs of role-subrole in the RBox.

# The *RBox Compatibility* service – behaviour

- If `Test 1` does not hold: warning that domain & range restrictions of either $R$ or $S$ are in conflict with the role hierarchy proposing either
  (i) To change the role hierarchy or
  (ii) To change domain & range restrictions or
  (iii) If the test on the domains fails, then propose a new axiom $R \sqsubseteq D'_R \times R_R$, where $D'_R \equiv D_R \sqcap D_S$[2], which subsequently has to go through the RBox compatibility service (and similarly when `Test 1` fails on range restrictions).

[2]The axiom $C_1 \equiv C_2$ is a shortcut for the axioms: $C_1 \sqsubseteq C_2$ and $C_2 \sqsubseteq C_1$.

# The *RBox Compatibility* service – behaviour

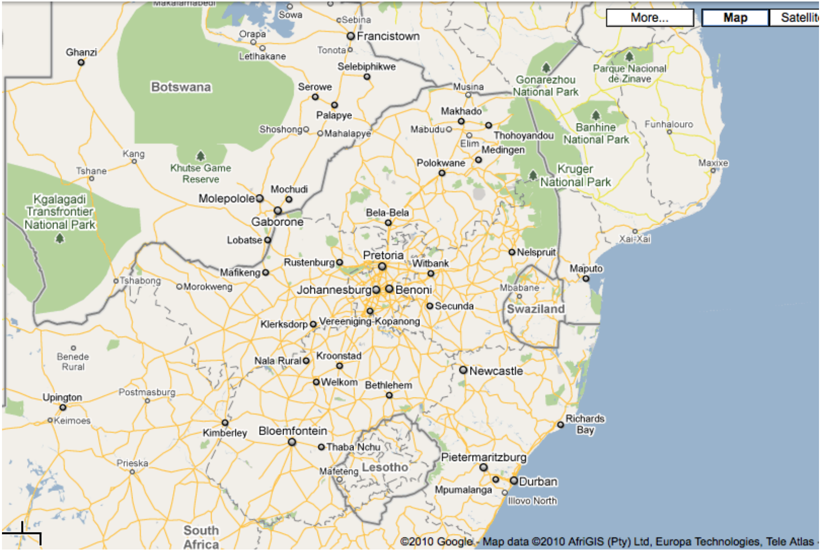
- If `Test 2` and `Test 3` fail: warn that $R$ cannot be a proper subrole of $S$ but that the two roles can be equivalent. Then, either:
  (a) Accept the possible equivalence between the two roles or
  (b) Change domain & range restrictions.
- Ignoring all warnings is allowed, too

# Outline

# Extensions in various directions

- Mereotopology, with location, GIS, Region Connection Calculus (http://www.comp.leeds.ac.uk/qsr/rcc.html)
- Mereogeometry
- Mereology and/vs granularity
- Temporal aspects of part-whole relations

# Knowledge and Google & AfriGIS



# Knowledge and Google & AfriGIS

- How can we represent
  - The Kruger Park *overlaps* with South Africa
  - Durban is a *tangential proper part* of South Africa
  - Gauteng is a *non-tangential proper part* of South Africa
  - Botswana is *connected to* South Africa (do they *share* a border?)
  - Lesotho is *spatially located within* the area of South Africa (but not part of)?
- Can we do all that with mereology? Use only spatial relations? Combining mereo+spatial?

# Knowledge and Google & AfriGIS

- How can we represent
  - The Kruger Park *overlaps* with South Africa
  - Durban is a *tangential proper part* of South Africa
  - Gauteng is a *non-tangential proper part* of South Africa
  - Botswana is *connected to* South Africa (do they *share* a border?)
  - Lesotho is *spatially located within* the area of South Africa (but not part of)?
- Can we do all that with mereology? Use only spatial relations? Combining mereo+spatial?

## Mereology with spatial notions

- Another primitive: $C$onnected, which is reflexive and symmetric
- More and more expressive theories, e.g.:
  - T: $C(x,x)$ and $C(x,y) \rightarrow C(y,x)$
  - MT: T and $P(x,y) \rightarrow E(x,y)$ where $E$ is enclosure $(E(x,y) =_{def} \forall z(C(z,x) \rightarrow C(z,y)))$
- Two primitives, $P$ and $C$, or *part* in terms of $C$?
  - $P =_{def} \forall z(C(z,x) \rightarrow C(z,y))$
- or perhaps "x and y are connected parts of z" as primitive, $CP(x,y,z)$, then:
  $P(x,y) =_{def} \exists z\ CP(x,z,y)$ and
  $C(x,y) =_{def} \exists z\ CP(x,y,z)$

# Mereology with spatial notions

- Another primitive: *Connected*, which is reflexive and symmetric
- More and more expressive theories, e.g.:
  - T: $C(x,x)$ and $C(x,y) \rightarrow C(y,x)$
  - MT: T and $P(x,y) \rightarrow E(x,y)$ where $E$ is enclosure ($E(x,y) =_{def} \forall z(C(z,x) \rightarrow C(z,y))$)
- Two primitives, $P$ and $C$, or *part* in terms of $C$?
  - $P =_{def} \forall z(C(z,x) \rightarrow C(z,y))$
- or perhaps "x and y are connected parts of z" as primitive, $CP(x,y,z)$, then: $P(x,y) =_{def} \exists z\ CP(x,z,y)$ and $C(x,y) =_{def} \exists z\ CP(x,y,z)$

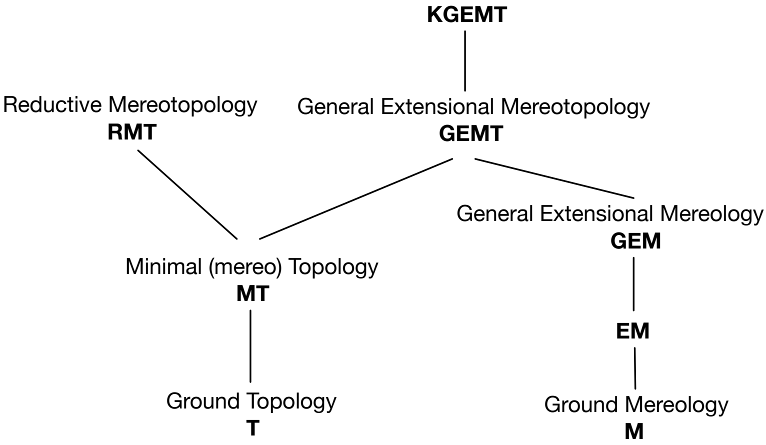
# Mereology with spatial notions

- Another primitive: $C$onnected, which is reflexive and symmetric
- More and more expressive theories, e.g.:
  - T: $C(x, x)$ and $C(x, y) \rightarrow C(y, x)$
  - MT: T and $P(x, y) \rightarrow E(x, y)$ where $E$ is enclosure $(E(x, y) =_{def} \forall z(C(z, x) \rightarrow C(z, y)))$
- Two primitives, $P$ and $C$, or *part* in terms of $C$?
  - $P =_{def} \forall z(C(z, x) \rightarrow C(z, y))$
- or perhaps "x and y are connected parts of z" as primitive, $CP(x, y, z)$, then:
  $P(x, y) =_{def} \exists z\ CP(x, z, y)$ and
  $C(x, y) =_{def} \exists z\ CP(x, y, z)$

## Mereology with spatial notions

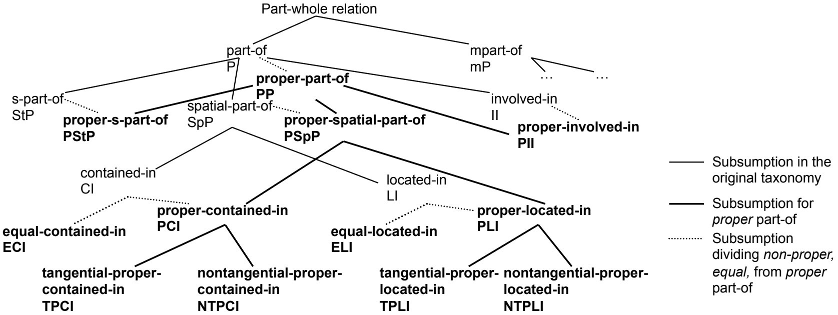
- Another primitive: $C$onnected, which is reflexive and symmetric
- More and more expressive theories, e.g.:
  - T: $C(x, x)$ and $C(x, y) \rightarrow C(y, x)$
  - MT: T and $P(x, y) \rightarrow E(x, y)$ where $E$ is enclosure $(E(x, y) =_{def} \forall z(C(z, x) \rightarrow C(z, y)))$
- Two primitives, $P$ and $C$, or *part* in terms of $C$?
  - $P =_{def} \forall z(C(z, x) \rightarrow C(z, y))$
- or perhaps "x and y are connected parts of z" as primitive, $CP(x, y, z)$, then:
  $P(x, y) =_{def} \exists z\ CP(x, z, y)$ and
  $C(x, y) =_{def} \exists z\ CP(x, y, z)$

# Some of the mereo- and topological theories




*Note: one can add explicit variations with Atom/Atomless and Boundary/Boundaryless*

# Extension to the taxonomy of part-whole relations



# Extension to the taxonomy of part-whole relations

$$\forall x, y \quad (ECI(x, y) \equiv CI(x, y) \wedge P(y, x) \tag{18}$$

$$\forall x, y \quad (PCI(x, y) \equiv PPO(x, y) \wedge R(x) \wedge R(y) \wedge \exists z, w(has\_3D(z, x) \wedge has\_3D(w, y) \wedge ED(z) \wedge ED(w))) \tag{19}$$

$$\forall x, y \quad (NTPCI(x, y) \equiv PCI(x, y) \wedge \forall z(C(z, x) \rightarrow O(z, y))) \tag{20}$$

$$\forall x, y \quad (TPCI(x, y) \equiv PCI(x, y) \wedge \neg NTPCI(x, y)) \tag{21}$$

$$\forall x, y \quad (ELI(x, y) \equiv LI(x, y) \wedge P(y, x) \tag{22}$$

$$\forall x, y \quad (PLI(x, y) \equiv PPO(x, y) \wedge R(x) \wedge R(y) \wedge \exists z, w(has\_2D(z, x) \wedge has\_2D(w, y) \wedge ED(z) \wedge ED(w))) \tag{23}$$

$$\forall x, y \quad (NTPLI(x, y) \equiv PLI(x, y) \wedge \forall z(C(z, x) \rightarrow O(z, y))) \tag{24}$$

$$\forall x, y \quad (TPLI(x, y) \equiv PLI(x, y) \wedge \neg NTPLI(x, y)) \tag{25}$$

# Implementability

- KGEMT requires second order logic
- No definitions of relations in OWL
- Recollect object property characteristics in the different OWL species
- What do we loose exactly regarding representation and, consequently, reasoning?
- See ESWC'12 paper

# Implementability

- KGEMT requires second order logic
- No definitions of relations in OWL
- Recollect object property characteristics in the different OWL species
- What do we loose exactly regarding representation and, consequently, reasoning?
- See ESWC'12 paper

# Outline

1. Parts, mereology, meronymy
   - Introduction
   - Mereology
   - Implementation
   - Meronymy
2. Taxonomy of types of part-whole relations
   - The taxonomy
   - Using the taxonomy of part-whole relations
   - RBox Compatibility
3. Extensions
4. Ontology Design Patterns

# Rationale

- It is hard to reuse only the “useful pieces” of a comprehensive (foundational) ontology, and the cost of reuse may be higher than developing a new ontology from scratch
- Need for small (or cleverly modularized) ontologies with explicit documentation of design rationales, and best reengineering practices
- Hence, in analogy to software design patterns: **ontology design patterns**
- ODPs summarize the good practices to be applied within design solutions
- ODPs keep track of the design rationales that have motivated their adoption

*content of slides based on Presutti et al, 2008*

# Rationale

- It is hard to reuse only the “useful pieces” of a comprehensive (foundational) ontology, and the cost of reuse may be higher than developing a new ontology from scratch
- Need for small (or cleverly modularized) ontologies with explicit documentation of design rationales, and best reengineering practices
- Hence, in analogy to software design patterns: **ontology design patterns**
- ODPs summarize the good practices to be applied within design solutions
- ODPs keep track of the design rationales that have motivated their adoption

*content of slides based on Presutti et al, 2008*

# Rationale

- It is hard to reuse only the "useful pieces" of a comprehensive (foundational) ontology, and the cost of reuse may be higher than developing a new ontology from scratch
- Need for small (or cleverly modularized) ontologies with explicit documentation of design rationales, and best reengineering practices
- Hence, in analogy to software design patterns: **ontology design patterns**
- ODPs summarize the good practices to be applied within design solutions
- ODPs keep track of the design rationales that have motivated their adoption

*content of slides based on Presutti et al, 2008*

# Rationale

- It is hard to reuse only the “useful pieces” of a comprehensive (foundational) ontology, and the cost of reuse may be higher than developing a new ontology from scratch
- Need for small (or cleverly modularized) ontologies with explicit documentation of design rationales, and best reengineering practices
- Hence, in analogy to software design patterns: **ontology design patterns**
- ODPs summarize the good practices to be applied within design solutions
- ODPs keep track of the design rationales that have motivated their adoption

*content of slides based on Presutti et al, 2008*

# Rationale

- It is hard to reuse only the "useful pieces" of a comprehensive (foundational) ontology, and the cost of reuse may be higher than developing a new ontology from scratch
- Need for small (or cleverly modularized) ontologies with explicit documentation of design rationales, and best reengineering practices
- Hence, in analogy to software design patterns: **ontology design patterns**
- ODPs summarize the good practices to be applied within design solutions
- ODPs keep track of the design rationales that have motivated their adoption

*content of slides based on Presutti et al, 2008*

# ODP definition

- An ODP is an information object
- A design pattern schema is the description of an ODP, including the roles, tasks, and parameters needed in order to solve an ontology design issue
- *An ODP is a modeling solution to solve a recurrent ontology design problem. It is an Information Object that expresses a Design Pattern Schema (or skin) that can only be satisfied by DesignSolutions. Design solutions provide the setting for Ontology Elements that play some ElementRole(s) from the schema.* (Presutti et al, 2008)

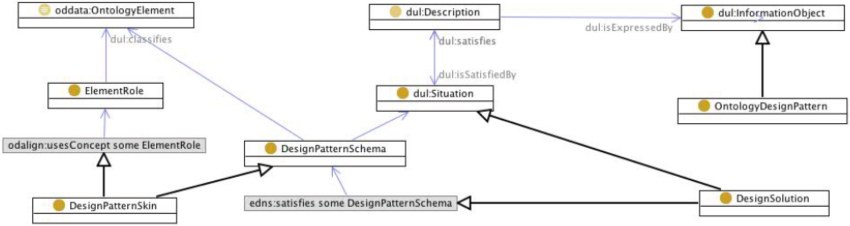
# ODP definition

- An ODP is an information object
- A design pattern schema is the description of an ODP, including the roles, tasks, and parameters needed in order to solve an ontology design issue
- *An ODP is a modeling solution to solve a recurrent ontology design problem. It is an Information Object that expresses a Design Pattern Schema (or skin) that can only be satisfied by DesignSolutions. Design solutions provide the setting for Ontology Elements that play some ElementRole(s) from the schema.* (Presutti et al, 2008)

# ODP definition

- An ODP is an information object
- A design pattern schema is the description of an ODP, including the roles, tasks, and parameters needed in order to solve an ontology design issue
- *An ODP is a modeling solution to solve a recurrent ontology design problem. It is an Information Object that expresses a Design Pattern Schema (or skin) that can only be satisfied by DesignSolutions. Design solutions provide the setting for Ontology Elements that play some ElementRole(s) from the schema.* (Presutti et al, 2008)

# ODP types


oddata:OntologyElement
dul:classifies
ElementRole
odalign:usesConcept some ElementRole
DesignPatternSkin
DesignPatternSchema
edns:satisfies some DesignPatternSchema
dul:Description
dul:satisfies
dul:isSatisfiedBy
dul:Situation
dul:isExpressedBy
dul:InformationObject
OntologyDesignPattern
DesignSolution

# Types of Patterns

- Six families of ODPs: Structural OPs, Correspondence OPs, Content OPs (CPs), Reasoning OPs, Presentation OPs, and Lexico-Syntactic OPs
- CPs can be distinguished in terms of the domain they represent
- Correspondence OPs (for reengineering and mappings—next lecture)
- Reasoning OPs are typical reasoning procedures
- Presentation OPs relate to ontology usability from a user perspective; e.g., we distinguish between Naming OPs and Annotation OPs
- Lexico-Syntactic OP are linguistic structures or schemas that permit to generalize and extract some conclusions about the meaning they express

# Types of Patterns

- Six families of ODPs: Structural OPs, Correspondence OPs, Content OPs (CPs), Reasoning OPs, Presentation OPs, and Lexico-Syntactic OPs
- CPs can be distinguished in terms of the domain they represent
- Correspondence OPs (for reengineering and mappings—next lecture)
- Reasoning OPs are typical reasoning procedures
- Presentation OPs relate to ontology usability from a user perspective; e.g., we distinguish between Naming OPs and Annotation OPs
- Lexico-Syntactic OP are linguistic structures or schemas that permit to generalize and extract some conclusions about the meaning they express

# Types of Patterns

- Six families of ODPs: Structural OPs, Correspondence OPs, Content OPs (CPs), Reasoning OPs, Presentation OPs, and Lexico-Syntactic OPs
- CPs can be distinguished in terms of the domain they represent
- Correspondence OPs (for reengineering and mappings—next lecture)
- Reasoning OPs are typical reasoning procedures
- Presentation OPs relate to ontology usability from a user perspective; e.g., we distinguish between Naming OPs and Annotation OPs
- Lexico-Syntactic OP are linguistic structures or schemas that permit to generalize and extract some conclusions about the meaning they express

# Types of Patterns

- Six families of ODPs: Structural OPs, Correspondence OPs, Content OPs (CPs), Reasoning OPs, Presentation OPs, and Lexico-Syntactic OPs
- CPs can be distinguished in terms of the domain they represent
- Correspondence OPs (for reengineering and mappings—next lecture)
- Reasoning OPs are typical reasoning procedures
- Presentation OPs relate to ontology usability from a user perspective; e.g., we distinguish between Naming OPs and Annotation OPs
- Lexico-Syntactic OP are linguistic structures or schemas that permit to generalize and extract some conclusions about the meaning they express

# Types of Patterns

- Six families of ODPs: Structural OPs, Correspondence OPs, Content OPs (CPs), Reasoning OPs, Presentation OPs, and Lexico-Syntactic OPs
- CPs can be distinguished in terms of the domain they represent
- Correspondence OPs (for reengineering and mappings—next lecture)
- Reasoning OPs are typical reasoning procedures
- Presentation OPs relate to ontology usability from a user perspective; e.g., we distinguish between Naming OPs and Annotation OPs
- Lexico-Syntactic OP are linguistic structures or schemas that permit to generalize and extract some conclusions about the meaning they express

# Types of Patterns

- Six families of ODPs: Structural OPs, Correspondence OPs, Content OPs (CPs), Reasoning OPs, Presentation OPs, and Lexico-Syntactic OPs
- CPs can be distinguished in terms of the domain they represent
- Correspondence OPs (for reengineering and mappings—next lecture)
- Reasoning OPs are typical reasoning procedures
- Presentation OPs relate to ontology usability from a user perspective; e.g., we distinguish between Naming OPs and Annotation OPs
- Lexico-Syntactic OP are linguistic structures or schemas that permit to generalize and extract some conclusions about the meaning they express

# Structural OPs

- Logical OPs:
  - Are compositions of logical constructs that solve a problem of expressivity in OWL-DL (and, in cases, also in OWL 2 DL)
  - Only expressed in terms of a logical vocabulary, because their signature (the set of predicate names, e.g. the set of classes and properties in an OWL ontology) is empty
  - Independent from a specific domain of interest
  - **Logical macros** compose OWL DL constructs; e.g. the universal+existential OWL macro
  - **Transformation patterns** translate a logical expression from a logical language into another; e.g. n-aries

# Structural OPs

- Logical OPs:
  - Are compositions of logical constructs that solve a problem of expressivity in OWL-DL (and, in cases, also in OWL 2 DL)
  - Only expressed in terms of a logical vocabulary, because their signature (the set of predicate names, e.g. the set of classes and properties in an OWL ontology) is empty
  - Independent from a specific domain of interest
  - **Logical macros** compose OWL DL constructs; e.g. the universal+existential OWL macro
  - **Transformation patterns** translate a logical expression from a logical language into another; e.g. n-aries

# Structural OPs

- Logical OPs:
  - Are compositions of logical constructs that solve a problem of expressivity in OWL-DL (and, in cases, also in OWL 2 DL)
  - Only expressed in terms of a logical vocabulary, because their signature (the set of predicate names, e.g. the set of classes and properties in an OWL ontology) is empty
  - Independent from a specific domain of interest
  - **Logical macros** compose OWL DL constructs; e.g. the universal+existential OWL macro
  - **Transformation patterns** translate a logical expression from a logical language into another; e.g. n-aries

# Structural OPs

- Logical OPs:
  - Are compositions of logical constructs that solve a problem of expressivity in OWL-DL (and, in cases, also in OWL 2 DL)
  - Only expressed in terms of a logical vocabulary, because their signature (the set of predicate names, e.g. the set of classes and properties in an OWL ontology) is empty
  - Independent from a specific domain of interest
  - **Logical macros** compose OWL DL constructs; e.g. the universal+existential OWL macro
  - **Transformation patterns** translate a logical expression from a logical language into another; e.g. n-aries

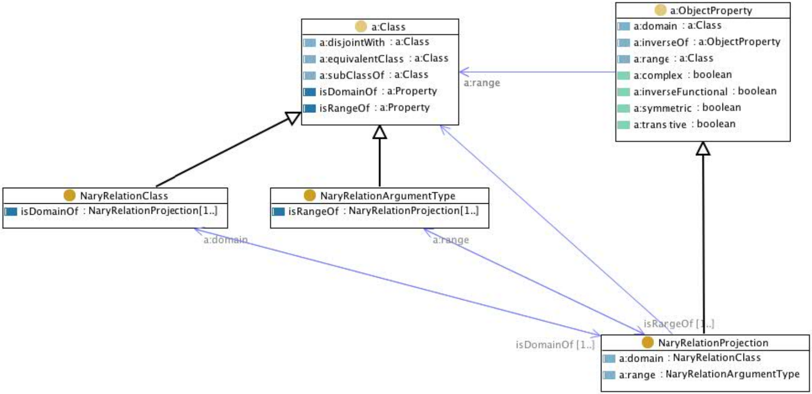
# Structural OPs

- Logical OPs:
  - Are compositions of logical constructs that solve a problem of expressivity in OWL-DL (and, in cases, also in OWL 2 DL)
  - Only expressed in terms of a logical vocabulary, because their signature (the set of predicate names, e.g. the set of classes and properties in an OWL ontology) is empty
  - Independent from a specific domain of interest
  - **Logical macros** compose OWL DL constructs; e.g. the universal+existential OWL macro
  - **Transformation patterns** translate a logical expression from a logical language into another; e.g. n-aries

# Structural OPs

- Logical OPs:
  - Are compositions of logical constructs that solve a problem of expressivity in OWL-DL (and, in cases, also in OWL 2 DL)
  - Only expressed in terms of a logical vocabulary, because their signature (the set of predicate names, e.g. the set of classes and properties in an OWL ontology) is empty
  - Independent from a specific domain of interest
  - **Logical macros** compose OWL DL constructs; e.g. the universal+existential OWL macro
  - **Transformation patterns** translate a logical expression from a logical language into another; e.g. n-aries

# Example: n-ary relation Logical OP



# Architectural OPs

- Architectural OPs are defined in terms of composition of Logical OPs that are used in order to affect the overall shape of the ontology; i.e., an Architectural OP identifies a composition of Logical OPs that are to be exclusively used in the design of an ontology
- Examples of Architectural OPs are: Taxonomy, Modular Architecture, and Lightweight Ontology
- E.g., **Modular Architecture** Architectural OP consists of an ontology network, where the involved ontologies play the role of modules, which are connected by the *owl:import* operation with one root ontology that imports all the modules

# Architectural OPs

- Architectural OPs are defined in terms of composition of Logical OPs that are used in order to affect the overall shape of the ontology; i.e., an Architectural OP identifies a composition of Logical OPs that are to be exclusively used in the design of an ontology
- Examples of Architectural OPs are: Taxonomy, Modular Architecture, and Lightweight Ontology
- E.g., **Modular Architecture** Architectural OP consists of an ontology network, where the involved ontologies play the role of modules, which are connected by the *owl:import* operation with one root ontology that imports all the modules

# Architectural OPs

- Architectural OPs are defined in terms of composition of Logical OPs that are used in order to affect the overall shape of the ontology; i.e., an Architectural OP identifies a composition of Logical OPs that are to be exclusively used in the design of an ontology
- Examples of Architectural OPs are: Taxonomy, Modular Architecture, and Lightweight Ontology
- E.g., **Modular Architecture** Architectural OP consists of an ontology network, where the involved ontologies play the role of modules, which are connected by the *owl:import* operation with one root ontology that imports all the modules

# Lexico-Syntactic OPs

- linguistic structures or schemas that consist of certain types of words following a specific order and that permit to generalize and extract some conclusions about the meaning they express; *verbalisation* patterns
- E.g., "subClassOf" relation, NP<subclass> be NP<superclass>, a Noun Phrase should appear before the verb—represented by its basic form or lemma, be in this example—and the verb should in its turn be followed by another Noun Phrase
- Other Lexical OPs provided for OWL's equivalence between classes, object property, subpropertyOf relation, datatype property, existential restriction, universal restriction, disjointness, union of classes
- Mainly for English language only, thus far
- Similar to idea of ORM's verbalization templates

# Lexico-Syntactic OPs

- linguistic structures or schemas that consist of certain types of words following a specific order and that permit to generalize and extract some conclusions about the meaning they express; *verbalisation* patterns
- E.g., "subClassOf" relation, NP<subclass> be NP<superclass>, a Noun Phrase should appear before the verb—represented by its basic form or lemma, be in this example—and the verb should in its turn be followed by another Noun Phrase
- Other Lexical OPs provided for OWL's equivalence between classes, object property, subpropertyOf relation, datatype property, existential restriction, universal restriction, disjointness, union of classes
- Mainly for English language only, thus far
- Similar to idea of ORM's verbalization templates

# Lexico-Syntactic OPs

- linguistic structures or schemas that consist of certain types of words following a specific order and that permit to generalize and extract some conclusions about the meaning they express; *verbalisation* patterns
- E.g., "subClassOf" relation, NP<subclass> be NP<superclass>, a Noun Phrase should appear before the verb—represented by its basic form or lemma, be in this example—and the verb should in its turn be followed by another Noun Phrase
- Other Lexical OPs provided for OWL's equivalence between classes, object property, subpropertyOf relation, datatype property, existential restriction, universal restriction, disjointness, union of classes
- Mainly for English language only, thus far
- Similar to idea of ORM's verbalization templates

# Lexico-Syntactic OPs

- linguistic structures or schemas that consist of certain types of words following a specific order and that permit to generalize and extract some conclusions about the meaning they express; *verbalisation* patterns
- E.g., "subClassOf" relation, NP<subclass> be NP<superclass>, a Noun Phrase should appear before the verb—represented by its basic form or lemma, be in this example—and the verb should in its turn be followed by another Noun Phrase
- Other Lexical OPs provided for OWL's equivalence between classes, object property, subpropertyOf relation, datatype property, existential restriction, universal restriction, disjointness, union of classes
- Mainly for English language only, thus far
- Similar to idea of ORM's verbalization templates

# Lexico-Syntactic OPs

- linguistic structures or schemas that consist of certain types of words following a specific order and that permit to generalize and extract some conclusions about the meaning they express; *verbalisation* patterns
- E.g., "subClassOf" relation, NP<subclass> be NP<superclass>, a Noun Phrase should appear before the verb—represented by its basic form or lemma, be in this example—and the verb should in its turn be followed by another Noun Phrase
- Other Lexical OPs provided for OWL's equivalence between classes, object property, subpropertyOf relation, datatype property, existential restriction, universal restriction, disjointness, union of classes
- Mainly for English language only, thus far
- Similar to idea of ORM's verbalization templates

# Summary

1. Parts, mereology, meronymy
   - Introduction
   - Mereology
   - Implementation
   - Meronymy
2. Taxonomy of types of part-whole relations
   - The taxonomy
   - Using the taxonomy of part-whole relations
   - RBox Compatibility
3. Extensions
4. Ontology Design Patterns