Ontology Engineering

Lecture 2: Recent trends: methods and methodologies

Dr. C. Maria Keet

email: `mkeet@cs.uct.ac.za`

home: `http://www.meteck.org`

Department of Computer Science
University of Cape Town, South Africa

Foundations and recent trends on ontology engineering
Universitat Politècnica de Catalunya, 2017

Outline

Ontology development

- You have some experience with an ontology language and representing some knowledge, which was given to you
- How to come up with the whole ontology in the first place?
- What can, or should, you do when you have to develop your own ontology?

Ontology development

- You have some experience with an ontology language and representing some knowledge, which was given to you
- How to come up with the whole ontology in the first place?
- What can, or should, you do when you have to develop your own ontology?

$\Rightarrow$ Just like in software engineering, there are methods and methodologies to guide you through it

Ontology development

- You have some experience with an ontology language and representing some knowledge, which was given to you
- How to come up with the whole ontology in the first place?
- What can, or should, you do when you have to develop your own ontology?

⇒ Just like in software engineering, there are methods and methodologies to guide you through it

- Recall (L1): an ontology is a logical theory "plus some more"
- In this lecture, we look into that "some more"

Topics for this lecture

- **Methodologies**
- **Modelling**, or: ontology quality:

Topics for this lecture

- **Methodologies**
  - Most are coarse-grained, i.e., a macro-level, processual information systems perspective; they do not (yet) contain all the permutations at each step
  - The actual modelling, or *ontology authoring*, using micro-level guidelines and tools
  - Zooming in on Test-Driven Development of ontologies
- **Modelling**, or: ontology quality:

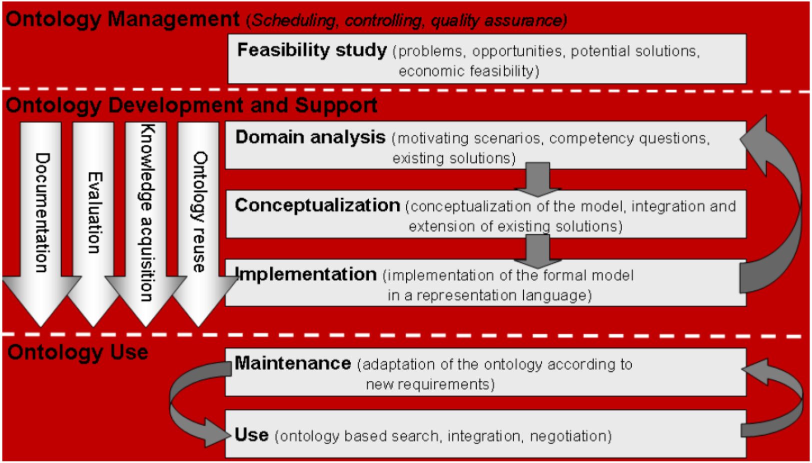
Topics for this lecture

- **Methodologies**
  - Most are coarse-grained, i.e., a macro-level, processual information systems perspective; they do not (yet) contain all the permutations at each step
  - The actual modelling, or *ontology authoring*, using micro-level guidelines and tools
  - Zooming in on Test-Driven Development of ontologies
- **Modelling**, or: ontology quality:
  - Methods assisting in modelling
  - Zooming in on modelling relations

Outline

1. Methodologies
   - Macro-level methodologies
   - Micro-level methodologies
   - Test-Driven Development
2. Modelling
   - Relationship issues
   - Semantics of relations
   - Some common relations
   - Reasoner-mediated modelling
3. Summary
   - Exercises

Typical stages of macro-level methodologies



(Source: Simperl et al., 2010)

A 'simple' methodology: METHONTOLOGY

- Basic methodological steps:
  - Specification: why, what are its intended uses, who are the prospective users
  - Conceptualization: with intermediate representations
  - Formalization: transforms the domain-expert understandable 'conceptual model' into a formal or semi-computable model
  - Implementation: represent it in an ontology language
  - Maintenance: corrections, updates, etc.

A ‘simple’ methodology: METHONTOLOGY

- Basic methodological steps:
  - Specification: why, what are its intended uses, who are the prospective users
  - Conceptualization: with intermediate representations
  - Formalization: transforms the domain-expert understandable ‘conceptual model’ into a formal or semi-computable model
  - Implementation: represent it in an ontology language
  - Maintenance: corrections, updates, etc.
- Additional tasks:
  - Management activities (schedule, control, and quality assurance)
  - Support activities (knowledge acquisition, integration, evaluation, documentation, and configuration management)

A 'simple' methodology: METHONTOLOGY

- Basic methodological steps:
  - Specification: why, what are its intended uses, who are the prospective users
  - Conceptualization: with intermediate representations
  - Formalization: transforms the domain-expert understandable 'conceptual model' into a formal or semi-computable model
  - Implementation: represent it in an ontology language
  - Maintenance: corrections, updates, etc.
- Additional tasks:
  - Management activities (schedule, control, and quality assurance)
  - Support activities (knowledge acquisition, integration, evaluation, documentation, and configuration management)
- Applied to chemical, legal domain, and others (More comprehensive assessment of extant methodologies in Corcho et al, 2003)

Tools to support this, and more

- Protégé, Topbraid, etc.
  - Standalone version with OWL 2 DL support, WebProtégé with a fragment of OWL 2
- MOdelling wiKI **MoKi** also has features that have become relevant more recently:
  - based on a *SemanticWiki*, used for collaborative and cooperative ontology development

Tools to support this, and more

- Protégé, Topbraid, etc.
  - Standalone version with OWL 2 DL support, WebProtégé with a fragment of OWL 2
- MOdelling wiKI **MoKi** also has features that have become relevant more recently:
  - based on a *SemanticWiki*, used for collaborative and cooperative ontology development
  - 'multi-modal access *at different levels of formality*: informal, semi-formal and formal (enables actors with different expertise contribute)

Extending the methodologies

- METHONTOLOGY and others (e.g., On-To-Knowledge, KACTUS approach) are methods for developing one *single* ontology
- Changing landscape in ontology development towards building "ontology networks"

Extending the methodologies

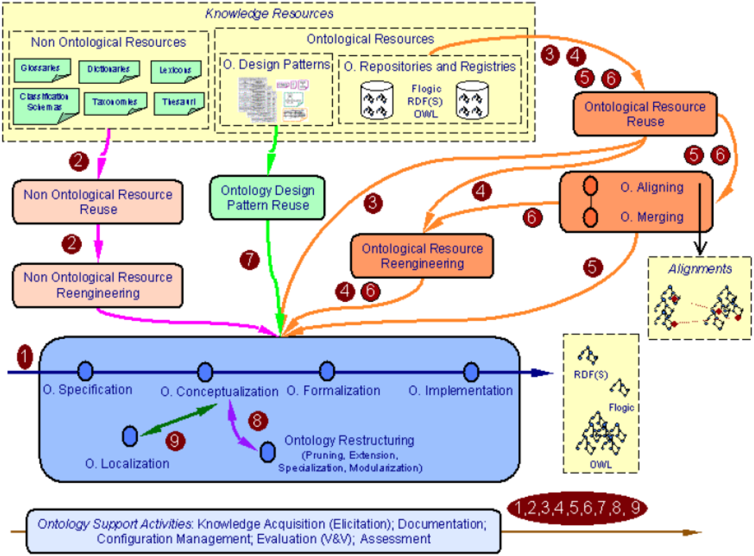
- METHONTOLOGY and others (e.g., On-To-Knowledge, KACTUS approach) are methods for developing one *single* ontology
- Changing landscape in ontology development towards building "ontology networks"
- Characteristics: dynamics, context, collaborative, distributed
- E.g., the NeOn methodology

Extending the methodologies: NeOn

- NeOn's "Glossary of Activities" identifies and defines 55 activities when ontology networks are collaboratively built
- Among others: ontology localization, -alignment, -formalization, -diagnosis, -enrichment etc.
- Divided into a matrix with "required" and "if applicable"
- Recognises there are several scenarios for ontology development, refining the typical monolithic 'waterfall' approach

(more info in neon_2008_d5.4.1.pdf)

Several scenarios for Building Ontology Networks



Micro-level methodologies

- Guidelines detailing how to go from informal to logic-based representations with instructions how to include the axioms and which ones are better than others
- To represent the formal and ontological details in an expressive ontology beyond just classes and some of their relationships so as to include guidance also for the axioms and ontological quality criteria

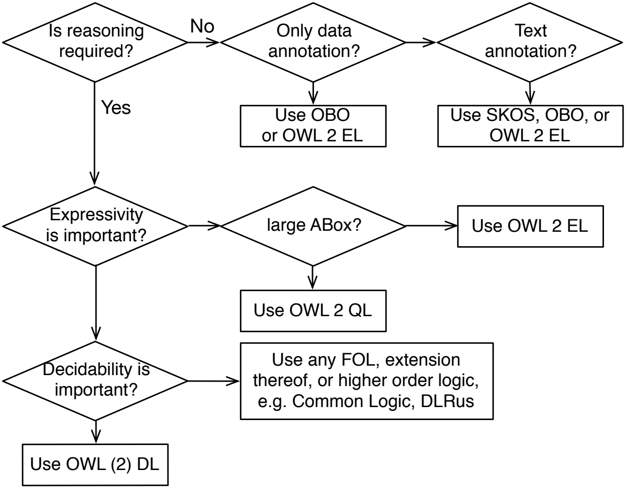
Micro-level methodologies

- Guidelines detailing how to go from informal to logic-based representations with instructions how to include the axioms and which ones are better than others
- To represent the formal and ontological details in an expressive ontology beyond just classes and some of their relationships so as to include guidance also for the axioms and ontological quality criteria
- Notably: OntoSpec, "Ontology development 101" (outdated!!!), DiDOn, TDD
- Methods & tools for sets of axioms; e.g.: advocatus diaboli, *SubProS* & *ProChainS*, ...

More detailed steps (generalised from DiDOn) (1/2)

1. Requirements analysis, regarding expressiveness (temporal, fuzzy, n-aries etc.), types of queries, reasoning services needed;
2. Design an ontology architecture, such as modular, and if so, in which way, distributed or not, etc.
3. Choose principal representation language and consider encoding peculiarities;

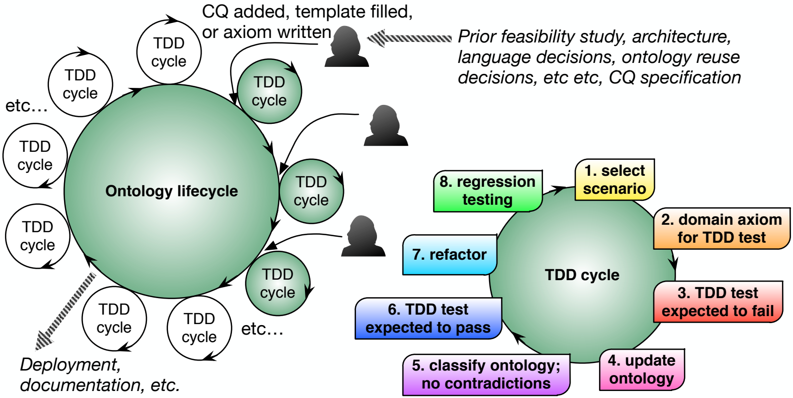
A few basic hints for choosing a language



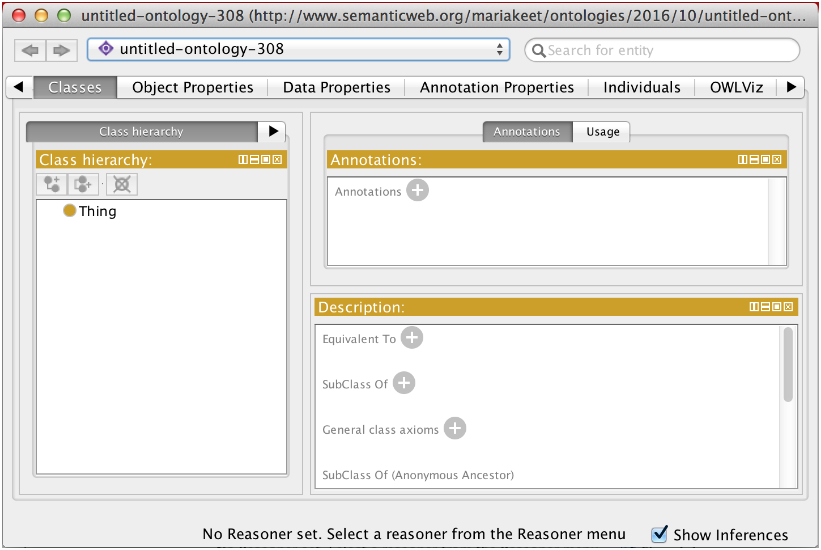
More detailed steps (generalised from DiDOn) (2/2)

4. Formalisation, including:
   - examine and add the classes, object properties, constraints, rules taking into account the imported ontologies;
   - use an automated reasoner for debugging/anomalous deductions;
   - use ontological reasoning services for quality checks (OntoClean, RBox Compatibility);
   - add annotations;
5. Generate versions in other ontology languages, 'lite' versions, etc, if applicable;

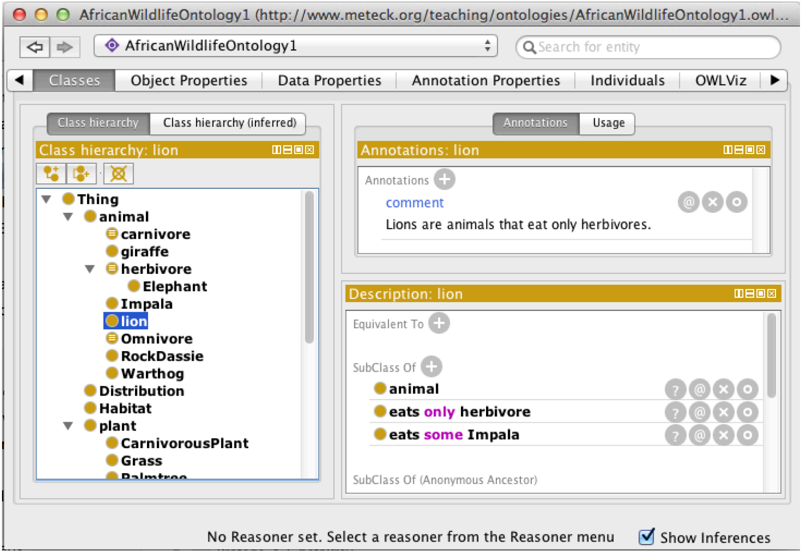
Test-driven Development



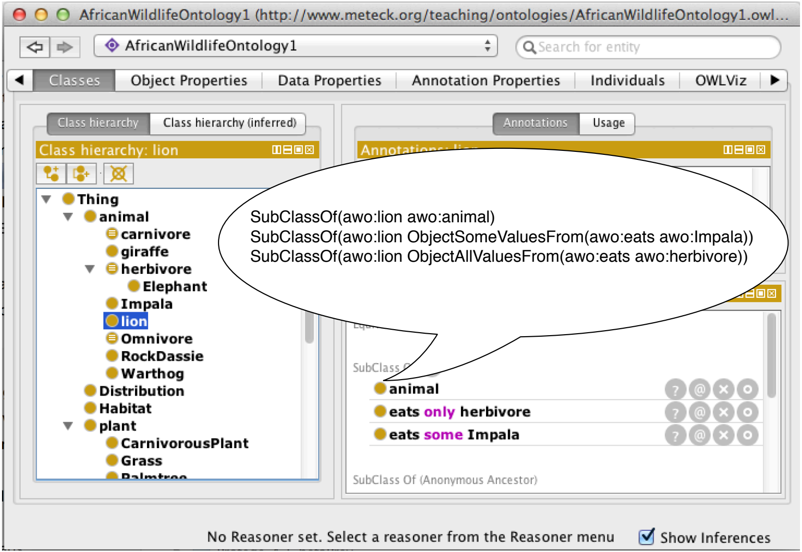
And then you open an ontology editor...



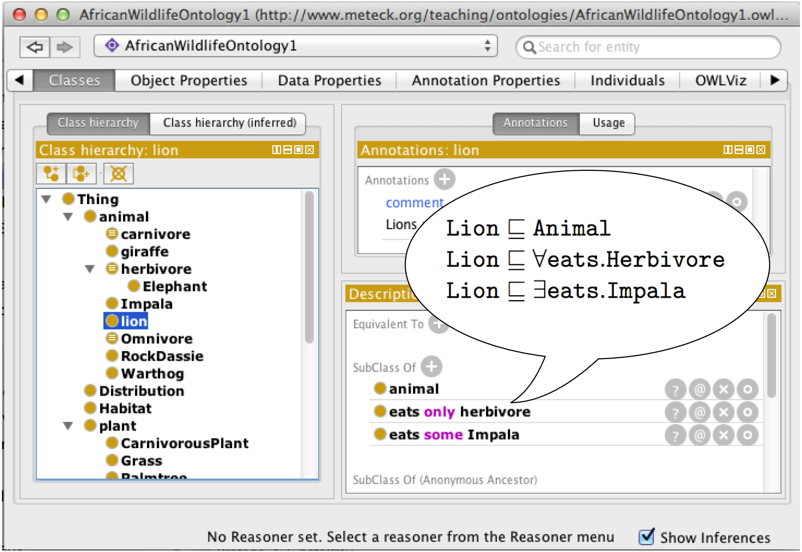
Or if you have something to start with:



Behind the facade



And behind that serialisation



Ontology development at the ‘micro-level’ level (cf. macro)

- We need to get those axioms into the ontology

Ontology development at the 'micro-level' level (cf. macro)

- We need to get those axioms into the ontology
- The actual modelling, or *ontology authoring*, using micro-level guidelines and tools
  - Methods, such as reverse engineering and text mining to start, OntoClean and OntoPartS to improve an ontology's quality
  - Parameters that affect ontology development, such as purpose, starting/legacy material, language
  - Tools to model, to reason, to debug, to integrate, to link to data

Ontology authoring

- Ontology authoring: on adding axioms to the Knowledge base

  Q1 "Does my ontology have axiom X?"
  - where X is, e.g., *all giraffes eat some twigs*
  - i.e., $Giraffe \sqsubseteq \exists eat.Twig$

- Current approaches:
  - For Q1: browsing, searching the *asserted* knowledge

Ontology authoring

- Ontology authoring: on adding axioms to the Knowledge base

  Q1 "Does my ontology have axiom X?"
  - where X is, e.g., *all giraffes eat some twigs*
  - i.e., $Giraffe \sqsubseteq \exists eat.Twig$

  Q2 "Will it still be consistent/class satisfiable if I add X?"
  - add, and try and see what the reasoner says about it
- Current approaches:
  - For Q1: browsing, searching the *asserted* knowledge
  - For Q2: essentially a *test-last* approach

Ontology authoring

- Ontology authoring: on adding axioms to the Knowledge base
  - Q1 "Does my ontology have axiom X?"
    - where X is, e.g., *all giraffes eat some twigs*
    - i.e., $Giraffe \sqsubseteq \exists eat.Twig$
  - Q2 "Will it still be consistent/class satisfiable if I add X?"
    - add, and try and see what the reasoner says about it
- Current approaches:
  - For Q1: browsing, searching the *asserted* knowledge
  - For Q2: essentially a *test-last* approach
- Cumbersome and time-consuming with larger ontologies
- Missing: a systematic testbed to do this in a methodical fashion
- It would need to relate to those macro-level processes

Addressing these issues

$\Rightarrow$ Reuse software engineering's notion of **Test-Driven Development**, based on **test-first**

(Recap) TDD in software development

- Methodology where one writes new code only if an automated test has failed [Beck(2004)].
- TDD permeates the whole development process
- TDD is a *test-first* approach rather than *test-last* (design, code, test) of unit tests
- More focussed, improves communication, improves understanding of required software behaviour, reduces design complexity [Kumar and Bansal(2013)]
- TDD produced code passes more externally defined tests—i.e, better software quality—and less time spent on debugging [Janzen(2005)]

Several scenarios for TDD (1/2)

I. CQ-driven TDD Specify Competency question, translate it into one or more axioms, which are the input of the relevant TDD test(s)
  - ex1 What software can perform task x?[1]
  - ex2 Given a data mining task/dataset, which of the valid or applicable workflows/algorithms will yield optimal results (or at least better results than the others)?
  - ex3 Are there learning algorithms that I can use on high-dimensional data without having to go through preliminary dimensionality reduction?

[1] more here: https://softwareontology.wordpress.com/2011/04/01/user-sourced-competency-questions-for-software/; other two examples from [Keet et al.(2015b)]

Several scenarios for TDD (2/2)

II-a. Ontology authoring-driven TDD - the knowledge engineer who knows which axiom s/he wants to add, types it, which is then fed directly into the TDD system

Behind the usability interface, what gets sent to the TDD system is one or more axioms

Several scenarios for TDD (2/2)

II-a. Ontology authoring-driven TDD - the knowledge engineer who knows which axiom s/he wants to add, types it, which is then fed directly into the TDD system

II-b. Ontology authoring-driven TDD - the domain expert uses a template or "logical macro" ODP [Presutti et al.(2008)], which map onto generic tests; e.g.:

- the all-some template, i.e., an axiom of the form $C \sqsubseteq \exists R.D$
- instantiate with relevant domain entities; e.g., $Professor \sqsubseteq \exists teaches.Course$
- the TDD test for the $C \sqsubseteq \exists R.D$ type of axiom is then run automatically

Behind the usability interface, what gets sent to the TDD system is one or more axioms

Tests in ontology engineering

- Early explorative work borrowing notion of testing [Vrandečić and Gangemi(2006)]—no framework, testbed
- CQs: patterns [Ren et al.(2014)], formalise into SPARQL queries—what, not how
- Instance-oriented approaches [Garca-Ramos et al.(2009), Kontokostas et al.(2014)], eXtreme Design NeON plugin, ODP rapid design [Blomqvist et al.(2012), Presutti et al.(2009)], RapidOWL [Auer(2006)]
- Tests for particular types of axioms:
    - disjointness [Ferré and Rudolph(2012)]
    - adding part-whole relation based domain and range constraints [Keet et al.(2013b)]

Tests in ontology engineering

- Tawny-Owl's subsumption tests [Warrender and Lord(2015)]. Tests tailored to the actual ontology rather than reusable 'templates' for the tests covering all OWL language features
- SCONE, BDD, focussing on natural language and examples, Cucumber at the back (F. Neuhaus, 2015)
- Methodologies:
  - none of the 9 methodologies reviewed by [Garcia et al.(2010)] are TDD-based
  - The Agile-inspired OntoMaven [Paschke and Schaefermeier(2015)] has OntoMvnTest with 'test cases' only for the usual syntax checking, consistency, and entailment

Tests in ontology engineering

- Full TDD ontology engineering [Keet and Ławrynowicz(2016), Ławrynowicz and Keet(2016), Davies et al.(2017)]
- Idea of unit tests has been proposed, there is a dearth of actual specifications as to what exactly is, or should be, going on in such as test
- No regression testing to check that perhaps an earlier modelled CQ—and thus a passed test—conflicts with a later one

Basic idea of Test-Driven Development for an ontology

1. Require: domain axiom x of type X is to be added to the ontology; e.g., x may be Professor $\sqsubseteq \exists$teaches.Course, which has pattern $C \sqsubseteq \exists R.D$.
2. Check the vocabulary elements of x are in ontology O (itself a TDD test);
3. Run the TDD test:
   3.1 The first execution should fail (check $O \nvDash x$ or not present)
   3.2 Update the ontology (add x), and
   3.3 Run the test again which then should pass (check that $O \models x$) and such that there is no new inconsistency or undesirable deduction
4. Run all previous successful tests, which still have to pass (i.e., regression testing); if not, resolve conflicting knowledge.

TDD test specification, preliminaries

- First iteration [Keet and Ławrynowicz(2016)]:
  - 42 test types for $\mathcal{SROIQ}$ covering *basic* axioms one can add to the TBox or RBox
  - Use the reasoner directly via OWL API
  - Notation of test in algorithm-style notation

TDD test specification, preliminaries

- First iteration [Keet and Ławrynowicz(2016)]:
  - 42 test types for $\mathcal{SROIQ}$ covering *basic* axioms one can add to the TBox or RBox
  - Use the reasoner directly via OWL API
  - Notation of test in algorithm-style notation
- Second iteration [Davies(2016), Davies et al.(2017)]:
  - TDD tests for general TBox axioms and some ABox assertions
  - Model for testing
  - More feedback (not just 'undefined', 'failed', 'OK')
  - Proofs

TDD test specification, preliminaries

- First iteration [Keet and Ławrynowicz(2016)]:
  - 42 test types for $\mathcal{SROIQ}$ covering *basic* axioms one can add to the TBox or RBox
  - Use the reasoner directly via OWL API
  - Notation of test in algorithm-style notation
- Second iteration [Davies(2016), Davies et al.(2017)]:
  - TDD tests for general TBox axioms and some ABox assertions
  - Model for testing
  - More feedback (not just ‘undefined’, ‘failed’, ‘OK’)
  - Proofs
- Third iteration: dealing with RBox inconsistencies [Keet(2012)], refactoring (ongoing)

Example: a TBox-test

Require: Test $T(C \sqsubseteq \exists R.D)$
$\alpha \leftarrow$ SubClassOf(?x ObjectSomeValuesFrom(R D))
if $C \notin \alpha$ **then** $\triangleright$ thus, $O \not\models C \sqsubseteq \exists R.D$
    return $T(C \sqsubseteq \exists R.D)$ is false
else
    return $T(C \sqsubseteq \exists R.D)$ is true
end if

Question: This is not the only possible design of a test. Name at least one other test design (i.e., irrespective of technologies) to test $T(C \sqsubseteq \exists R.D)$; e.g., to find out whether, say, $Lion \sqsubseteq \exists eats.Impala$ is entailed in the ontology

Revisiting the general idea of TDD for an ontology

1. Require: domain axiom x of type X is to be added to the ontology; e.g., x may be Professor $\sqsubseteq \exists$teaches.Course, which has pattern $C \sqsubseteq \exists R.D$.
2. Check the vocabulary elements of x are in ontology O (itself a TDD test);
3. Run the TDD test:
   3.1 The first execution should fail (check $O \nvDash x$ or not present)
   3.2 Update the ontology (add x), and
   3.3 Run the test again which then should pass (check that $O \models x$) and such that there is no new inconsistency or undesirable deduction
4. Run all previous successful tests, which still have to pass (i.e., regression testing); if not, resolve conflicting knowledge.

A model for testing–possible test results

- Ontology already inconsistent
- Ontology already incoherent: that is, one or more of its named classes are unsatisfiable.
- Missing entity in axiom: The axiom contains one or more named classes or properties which are not declared in the ontology.
- Axiom causes inconsistency
- Axiom causes incoherence
- Axiom absent: The axiom is not entailed by the ontology, but it could be added without negative consequences.
- Axiom entailed: The axiom is already entailed by the ontology

Formally

Definition

Given an ontology O which is consistent and coherent, and an axiom A such that $\Sigma(A) \subseteq \Sigma(O)$, the result of testing A against O is

$$\text{test}_O(A) = \begin{cases} \text{entailed} & \text{if } O \vdash A \\ \text{inconsistent} & \text{if } O \cup A \vdash \bot \\ \text{incoherent} & \text{if } O \cup A \nvdash \bot \\ & \wedge(\exists C \in \Sigma_C(O)) \text{ s.t. } O \cup A \vdash C \sqsubseteq \bot \\ \text{absent} & \text{otherwise} \end{cases}$$

Generalisation

Note: now C and D can be *any* class expression, not just only a named class

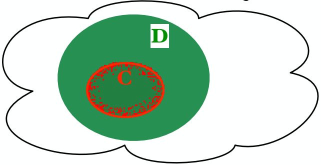
Algorithm 1 Function $\textsc{TestSubclassOf}(C, D)$

Input: C, D class expressions

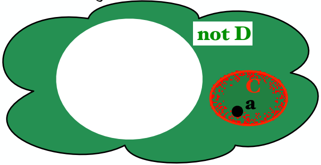
```
if GetInstances(C ⊓ ¬D) ≠ ∅ then
    return inconsistent
else if GetSubclasses(C ⊓ ¬D) ≠ ∅ then
    return incoherent
else if IsSatisfiable(C ⊓ ¬D) == false then
    return entailed
else
    return absent
end if
```

Graphically

We want to test whether this already holds in O.

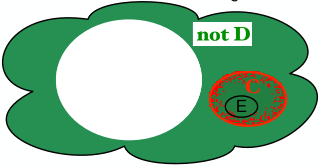


There is an object, **a**, that is a C and not a D...



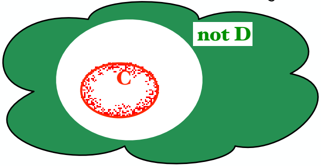
... so O with C is-a D would turn out to be inconsistent.

There is some class E subsumed by C and not a D...



... so C is-a D would cause O to be incoherent.

There cannot be a class E subsumed by C and not a D...



... so C is-a D is entailed already in O.

What remains: C is-a D is **absent**.

Design considerations and issues

- Which technology to use?
- DL Query tab possible
  - to cumbersome, not all tests possible
- SPARQL-OWL's implementation OWL-BGP and its SPARQL SELECT, SPARQL answering engine, and Hermit v1.3.8 [Kollia et al.(2011)]
  - Limited RBox tests (note: does not implement ASK queries)
- SPARQL-DL's implementation with its ASK queries
  - Limited RBox tests
- Use just the OWL API + a DL reasoner

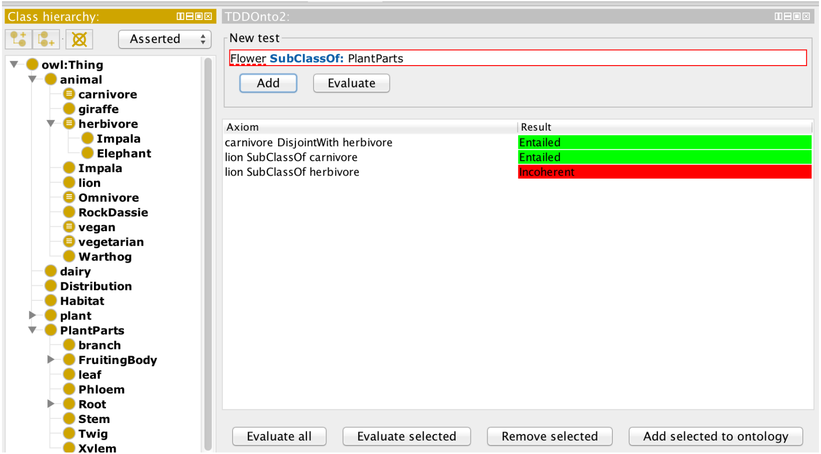
TDDOnto2

- TDDOnto tool as Protégé plugin
- Manages test specification and execution, ontology update
- ‘wraps’ around the actual execution of the test (SPARQL query, reasoner) for creation/deletion mock entities, the true/false returned

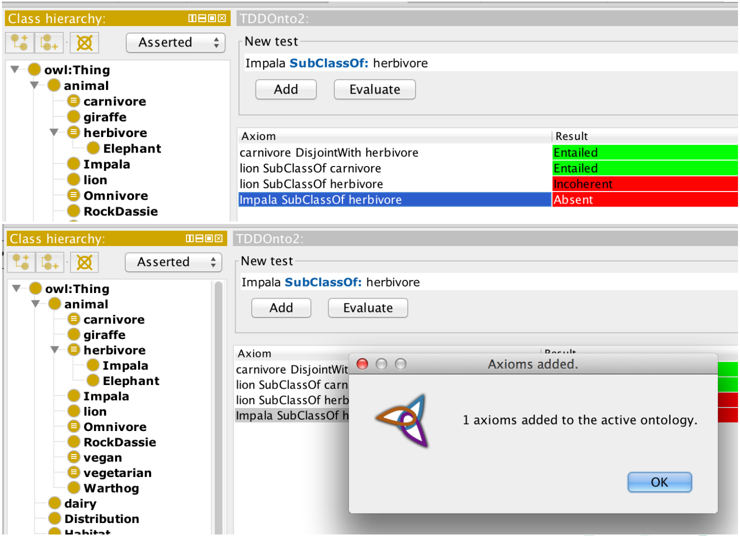
TDDOnto2

- TDDOnto tool as Protégé plugin
- Manages test specification and execution, ontology update
- ‘wraps’ around the actual execution of the test (SPARQL query, reasoner) for creation/deletion mock entities, the true/false returned
- To make a long story of performance evaluations short: TDDonto2 uses the reasoner, for it is the fastest of the options

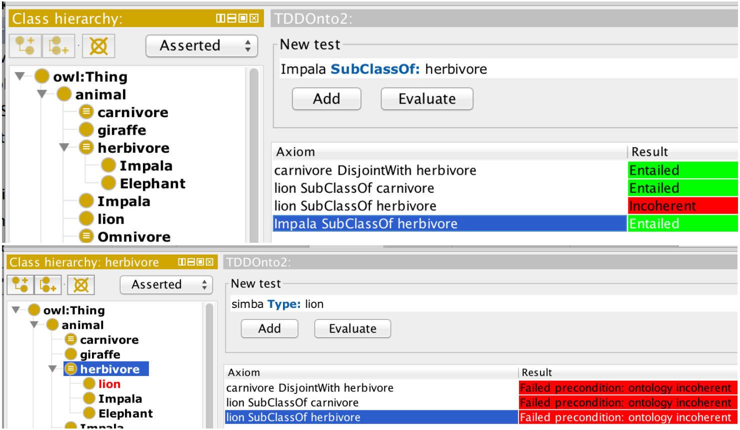
Screenshots



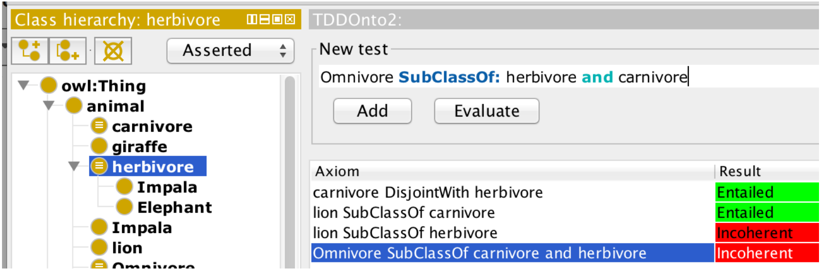
Screenshots

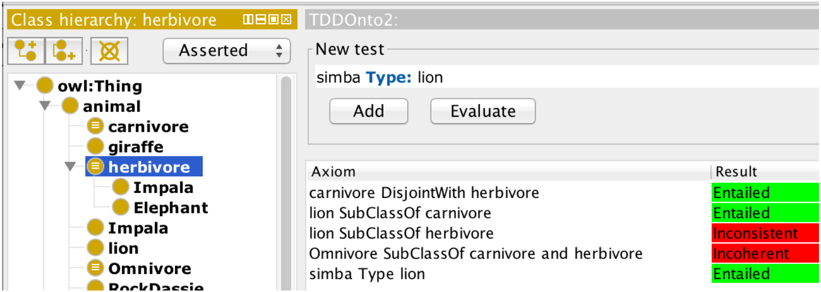


Screenshots

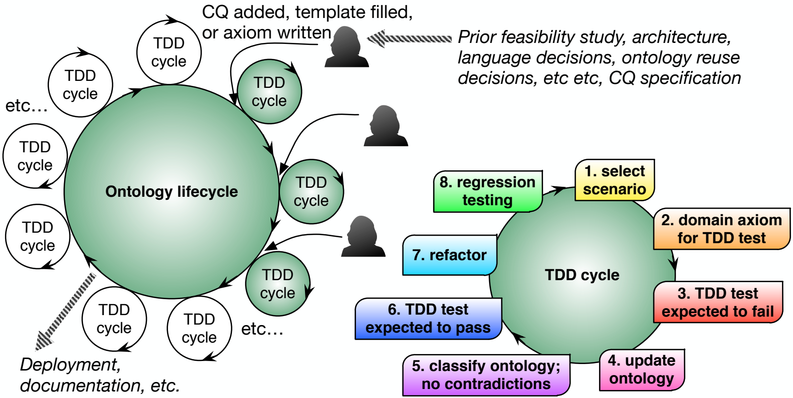


Screenshots

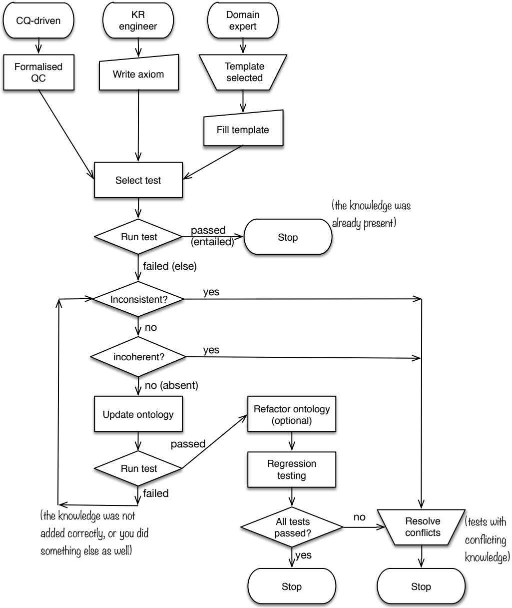




Methodology sketch



Test-Driven Development



Notes

- The picture shows the basic loop only

Notes

- The picture shows the basic loop only
- What if Step 5 goes wrong (inconsistent/incoherent ontology):
- What if after refactoring (step 7), regression (step 8) fails:

Notes

- The picture shows the basic loop only
- What if Step 5 goes wrong (inconsistent/incoherent ontology):
  - What is the source of the inconsistency?
  - Was it a previous test, hence, contradicting CQs?
- What if after refactoring (step 7), regression (step 8) fails:

Notes

- The picture shows the basic loop only
- What if Step 5 goes wrong (inconsistent/incoherent ontology):
  - What is the source of the inconsistency?
  - Was it a previous test, hence, contradicting CQs?
- What if after refactoring (step 7), regression (step 8) fails:
  - Is a previous test obsolete?
  - Error introduced in the refactoring?

Notes

- The picture shows the basic loop only
- What if Step 5 goes wrong (inconsistent/incoherent ontology):
  - What is the source of the inconsistency?
  - Was it a previous test, hence, contradicting CQs?
- What if after refactoring (step 7), regression (step 8) fails:
  - Is a previous test obsolete?
  - Error introduced in the refactoring?
- What does ‘refactoring’ an ontology mean anyway?
  - (we have some ideas, but too preliminary at this stage)

Outline

Preliminaries

- Domain experts are expert in their subject domain, which is not logic

Preliminaries

- Domain experts are expert in their subject domain, which is not logic
- Modellers often do not understand the subject domain well

Preliminaries

- Domain experts are expert in their subject domain, which is not logic
- Modellers often do not understand the subject domain well
- The more expressive the language, the easier it is to make errors or bump into unintended entailments

Preliminaries

- Domain experts are expert in their subject domain, which is not logic
- Modellers often do not understand the subject domain well
- The more expressive the language, the easier it is to make errors or bump into unintended entailments
- Simple languages can represent more than it initially may seem (by some more elaborate encoding), which clutters the ontology and affects comprehension

Preliminaries

- Domain experts are expert in their subject domain, which is not logic
- Modellers often do not understand the subject domain well
- The more expressive the language, the easier it is to make errors or bump into unintended entailments
- Simple languages can represent more than it initially may seem (by some more elaborate encoding), which clutters the ontology and affects comprehension
- In short: people make errors (w.r.t. their intentions) in the modelling task, and automated reasoners can help fix that

Preliminaries

- Using automated reasoners for 'debugging' ontologies, requires one to know about reasoning services
- Using standard reasoning services
- Reasoning services tailored to pinpointing the errors and explaining the entailments

Sampling of methods and tools for 'debugging'

- Finding pitfalls[2]: OntOlogy Pitfall Scanner! (OOPS!), and TIPS to prevent them [Poveda-Villalón et al.(2012), Keet et al.(2015a)]
- Finding errors and correcting them: *SubProS* and *ProChainS* [Keet(2012)]
- Preventing making errors: FORZA [Keet et al.(2013b)]
- When to declare classes disjoint: advocatus diaboli [Ferré and Rudolph(2012), Ferré(2016)]
- See also examples in the lecture notes, WoDOOM workshop proceedings

[2] pitfall: among others: undesirable deductions, inconsistent naming scheme, declaring a property transitive but with incompatible domain and range

Common errors

- Unsatisfiable classes
  - In the tools: the unsatisfiable classes end up as direct subclass of `owl:Nothing`
  - Sometimes one little error generates a whole cascade of unsatisfiable classes

Common errors

- Unsatisfiable classes
  - In the tools: the unsatisfiable classes end up as direct subclass of `owl:Nothing`
  - Sometimes one little error generates a whole cascade of unsatisfiable classes
- Satisfiability checking can cause rearrangement of the class tree and any inferred relationships to be associated with a class definition: 'desirable' vs. 'undesireable' inferred subsumptions

Common errors

- Unsatisfiable classes
  - In the tools: the unsatisfiable classes end up as direct subclass of `owl:Nothing`
  - Sometimes one little error generates a whole cascade of unsatisfiable classes
- Satisfiability checking can cause rearrangement of the class tree and any inferred relationships to be associated with a class definition: 'desirable' vs. 'undesireable' inferred subsumptions
- Inconsistent ontologies: all classes *taken together* unsatisfiable

Common errors

- Basic set of clashes for concepts (w.r.t. tableaux algorithms) are:
  - Atomic: An individual belongs to a class and its complement
  - Cardinality: An individual has a max cardinality restriction but is related to more distinct individuals
  - Datatype: A literal value violates the (global or local) range restrictions on a datatype property

Common errors

- Basic set of clashes for concepts (w.r.t. tableaux algorithms) are:
  - Atomic: An individual belongs to a class and its complement
  - Cardinality: An individual has a max cardinality restriction but is related to more distinct individuals
  - Datatype: A literal value violates the (global or local) range restrictions on a datatype property
- Basic set of clashes for KBs (ontology + instances) are:
  - Inconsistency of assertions about individuals, e.g., an individual is asserted to belong to disjoint classes or has a cardinality restriction but related to more individuals
  - Individuals related to unsatisfiable classes
  - Defects in class axioms involving nominals (`owl:oneOf`, if present in the language)

Setting

- Representing hierarchies of classes [/concepts/universals/entity types/...] typically received first/most/only attention
- Things become interesting from the viewpoint of automated reasoning only if there are other axioms, or: properties of those classes

Setting

- Representing hierarchies of classes [/concepts/universals/entity types/...] typically received first/most/only attention
- Things become interesting from the viewpoint of automated reasoning only if there are other axioms, or: properties of those classes

⇒ How to model those? (and have good quality)

⇒ What effect does that have on the deductions? (preferably desired ones)

Some problematic examples with relationships

A. Trans(partOf)
$\texttt{Hand} \sqsubseteq \exists \texttt{partOf.Musician}$
$\texttt{Musician} \sqsubseteq \exists \texttt{partOf.Orchestra}$
Deducing that each Hand is part of an Orchestra is *'wrong'*

Some problematic examples with relationships

A. Trans(partOf)
$\texttt{Hand} \sqsubseteq \exists \texttt{partOf}.\texttt{Musician}$
$\texttt{Musician} \sqsubseteq \exists \texttt{partOf}.\texttt{Orchestra}$
Deducing that each Hand is part of an Orchestra is *'wrong'*

B. $\texttt{hasMainTable} \circ \texttt{hasFeature} \sqsubseteq \texttt{hasFeature}$
$\texttt{hasMainTable} \sqsubseteq \texttt{DataSet} \times \texttt{DataTable}$
$\texttt{hasFeature} \sqsubseteq \texttt{DataTable} \times \texttt{Feature}$

Some problematic examples with relationships

A. $\mathtt{Trans(partOf)}$

$\mathtt{Hand} \sqsubseteq \exists \mathtt{partOf.Musician}$

$\mathtt{Musician} \sqsubseteq \exists \mathtt{partOf.Orchestra}$

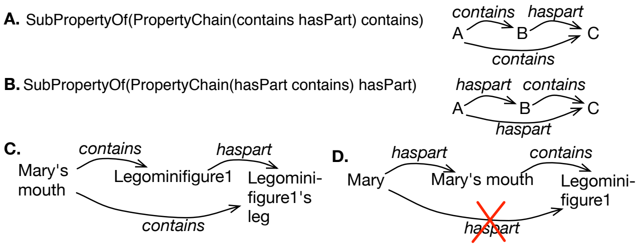
Deducing that each Hand is part of an Orchestra is *'wrong'*

B. $\mathtt{hasMainTable} \circ \mathtt{hasFeature} \sqsubseteq \mathtt{hasFeature}$

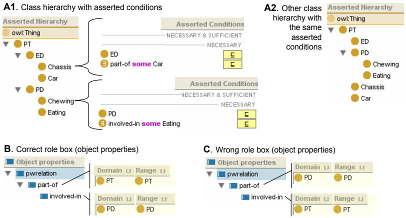
$\mathtt{hasMainTable} \sqsubseteq \mathtt{DataSet} \times \mathtt{DataTable}$

$\mathtt{hasFeature} \sqsubseteq \mathtt{DataTable} \times \mathtt{Feature}$

- Deduces $\mathtt{DataSet} \sqsubseteq \mathtt{DataTable}$, which is *'wrong'*

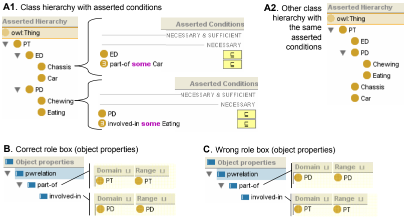


And old issue



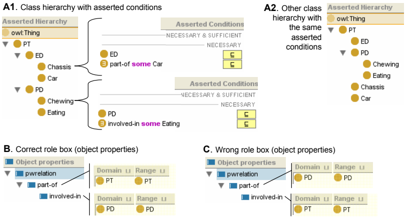
- *(Live with Protégé)*

And old issue



- *(Live with Protégé)*
- A1+B: OK; A2+B: OK
- A1+C: `Chassis` inconsistent; A2+C: `Chassis` (re)classified as a `PD`

And old issue



- *(Live with Protégé)*
- A1+B: OK; A2+B: OK
- A1+C: `Chassis` inconsistent; A2+C: `Chassis` (re)classified as a `PD`

C. But actually, the property hierarchy is *wrong* (mostly ignored by the DL/OWL reasoner, so can't find that mistake)

Other modelling and implementation issues

- **Poll**: are `teaches` and `taught by` two relations?

Other modelling and implementation issues

- **Poll**: are `teaches` and `taught by` two relations?
  - ⇒ differentiate between *relation* between entities and *relational expression* describing that state

Other modelling and implementation issues

- **Poll**: are `teaches` and `taught by` two relations?
  - ⇒ differentiate between *relation* between entities and *relational expression* describing that state
- **Poll**: How do you map UML's association ends (or ORM's roles) to an OWL object property (or vv.)?

Other modelling and implementation issues

- **Poll**: are `teaches` and `taught by` two relations?
  - ⇒ differentiate between *relation* between entities and *relational expression* describing that state
- **Poll**: How do you map UML's association ends (or ORM's roles) to an OWL object property (or vv.)?
  - ⇒ Bit tricky, you have to make a modelling decision...
- These two questions surface as a consequence of different ontological commitment as to what a relation really is (or what you're convinced of it is)

A few other modelling questions

- Should you introduce a minimum amount of properties in your ontology, or many?
- Always (try to) declare domain and range axioms?
- Use explicit inverses (extending the vocabulary) or not?
- What about ternaries?
- How to find and fix mistakes and pitfalls?
- What if solution X is better modelling than option Y but computationally more costly than Y?

Toward solving such issues

- Meaning of relations
  - Different modelling/representation languages have varying 'ontological commitments'
  - When a relation(ship) is a specialisation of another
- Reuse relations that are already investigated widely cf. reinventing the wheel
- Methods and tools to avoid pitfalls

Notes from philosophy

- Relations investigated in philosophy
  - Nature of relation itself (standard, positionalist, anti-positionalist) [Fine(2000), Leo(2008)]
  - Relationships as endurants or events [Guarino and Guizzardi(2015), Guarino and Guizzardi(2016)]
  - Nature and properties of some specific domain-independent relations (parthood, portions, participation, causation); see the Stanford Encyclopedia of Philosophy for gentle introductions
  - 'Categories' of relations (material, formal) (e.g., [Guizzardi and Wagner(2008)])
  - (among others)
- Some results more useful for ontologies and conceptual modelling than others, some even for tool development

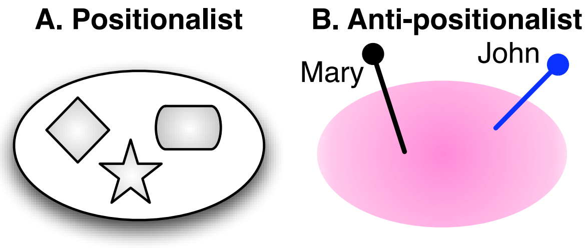
What relations are

- Three main options: standard, positionalist, anti-positionalist [Fine(2000), Leo(2008)]
- Applied to trying to resolve issues in metamodels, formalisations and tools [Keet(2009), Keet and Fillottrani(2015), Fillottrani and Keet(2015)]
- Not the arguments here, only present what they are

What relations are

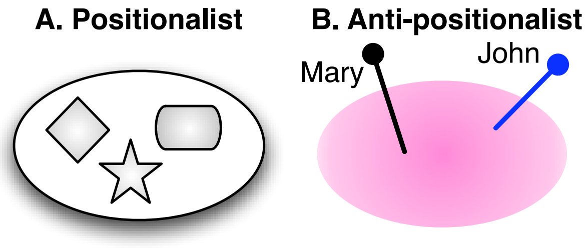
- Three main options: standard, positionalist, anti-positionalist [Fine(2000), Leo(2008)]
- Applied to trying to resolve issues in metamodels, formalisations and tools [Keet(2009), Keet and Fillottrani(2015), Fillottrani and Keet(2015)]
- Not the arguments here, only present what they are
- Standard view relies on linguistics and the English language in particular
- Formalisation predicate-centred, order of entities important

Graphical depictions positionalist, anti-positionalist



- Positionalist needs argument places in the "fundamental furniture of the universe", anti-positionalist does not

Graphical depictions positionalist, anti-positionalist



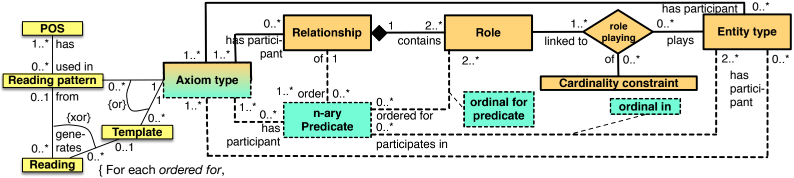
- Positionalist needs argument places in the "fundamental furniture of the universe", anti-positionalist does not
- UML Class Diagrams, ORM, ER all positionalist [Keet and Fillottrani(2015)], OWL, FOL, and most of DL take standard view

How to bridge 'standard view' with 'positionalist'?

- To at least achieve a faithful mapping between conceptual model and its formalisation in a standard view-based logic
- Also helps with linguistic annotations

How to bridge ‘standard view’ with ‘positionalist’?

- To at least achieve a faithful mapping between conceptual model and its formalisation in a standard view-based logic
- Also helps with linguistic annotations
- Model and formalisation of the mapping in [Keet and Chirema(2016)]



{ For each *ordered for*,
the *Roles ordered for* a *Predicate* are *contained in* a *Relationship* that is the *Relationship of* that *Predicate*.
(but a *Relationship* that *contains Roles* need not have a *Predicate*, and *Roles* need not be *ordered for* a *Predicate*)
Each *Axiom has participant* either a *Relationship* or an *Entity type* or both, or an *n-ary Predicate* or an *Entity type* or both. }

Questions and problems to address

- Modelling flaws in the RBox show up as unexpected or undesirable deductions regarding classes in the TBox, but current explanation algorithms (e.g., [Horridge et al.(2008), Parsia et al.(2005), Kalyanpur et al.(2006)]) mostly do not point to the actual flaw in the RBox
- What are the features of a 'good' RBox w.r.t. object property expressions?
- What type of flaws are being made?
- See [Keet(2014)]

Preliminaries (1/2)

- "basic form" for sub-properties, i.e., $S \sqsubseteq R$,
- "complex form" with property chains
- $R \sqsubseteq C_1 \times C_2$ as shortcut for domain and range axioms $\exists R \sqsubseteq C_1$ and $\exists R^- \sqsubseteq C_2$ where C_1 and C_2 are generic classes; `ObjectPropertyDomain(OPE CE)` and `ObjectPropertyRange(OPE CE)` in OWL.
- $R \sqsubseteq \top \times \top$ when no domain and range axiom has been declared

Definition (User-defined Domain and Range Classes)

Let R be an OWL object property and $R \sqsubseteq C_1 \times C_2$ its associated domain and range axiom. Then, with the symbol D_R we indicate the *User-defined Domain* of R—i.e., $D_R = C_1$—and with the symbol R_R we indicate the *User-defined Range* of R—i.e., $R_R = C_2$.

Definition ((Regular) Role Inclusion Axioms ([Horrocks et al.(2006)]))

Let $\prec$ be a regular order on roles. A **role inclusion axiom** (RIA for short) is an expression of the form $w \sqsubseteq R$, where w is a finite string of roles not including the universal role U, and $R \neq U$ is a role name. A **role hierarchy** $\mathcal{R}_h$ is a finite set of RIAs. An interpretation $\mathcal{I}$ **satisfies** a role inclusion axiom $w \sqsubseteq R$, written $\mathcal{I} \models w \sqsubseteq R$, if $w^{\mathcal{I}} \subseteq R^{\mathcal{I}}$. An interpretation is a **model** of a role hierarchy $\mathcal{R}_h$ if it satisfies all RIAs in $\mathcal{R}_h$, written $\mathcal{I} \models \mathcal{R}_h$. A RIA $w \sqsubseteq R$ is $\prec$**-regular** if R is a role name, and

$w = R \circ R$, or

$w = R^-$, or

$w = S_1 \circ \ldots \circ S_n$ and $S_i \prec R$, for all $1 \geq i \geq n$, or

$w = R \circ S_1 \circ \ldots \circ S_n$ and $S_i \prec R$, for all $1 \geq i \geq n$, or

$w = S_1 \circ \ldots \circ S_n \circ R$ and $S_i \prec R$, for all $1 \geq i \geq n$.

Finally, a role hierarchy $\mathcal{R}_h$ is **regular** if there exists a regular order $\prec$ such that each RIA in $\mathcal{R}_h$ is $\prec$-regular.

Object sub-properties

- Given $S \sqsubseteq R$, then all individuals in the property assertions involving property S must also be related to each other through property R (OWL 2 Spec.).
- Subsumption for OWL object properties (DL roles) holds if the subsumed property is more constrained such that in every model, the set of individual property assertions is a subset of those of its parent property

Object sub-properties

- Given $S \sqsubseteq R$, then all individuals in the property assertions involving property S must also be related to each other through property R (OWL 2 Spec.).
- Subsumption for OWL object properties (DL roles) holds if the subsumed property is more constrained such that in every model, the set of individual property assertions is a subset of those of its parent property
- Two ways to constrain a property, and either one suffices:
  - By specifying its domain or range
  - By declaring the property's characteristics

Constraining a property

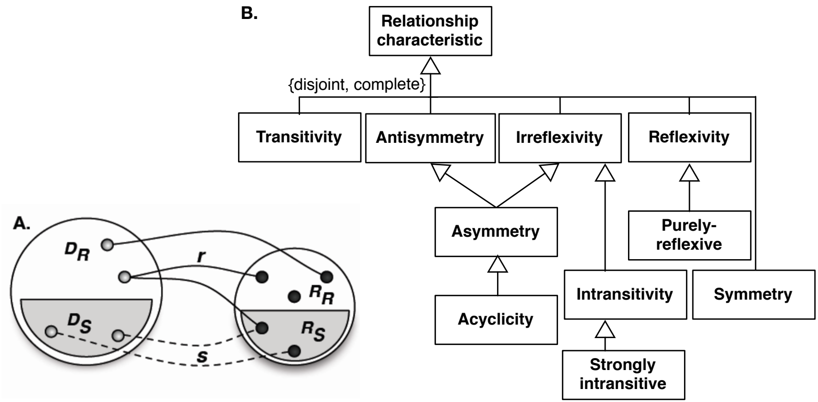


Figure: A: Example, alike the so-called 'subsetting' idea in UML; B: hierarchy of property characteristics (Based on Halpin 2001, 2011)

Constraining a property

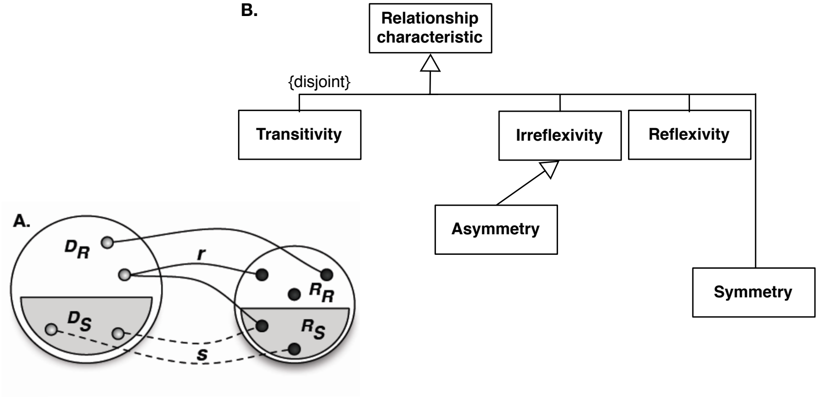


Figure: A: Example, alike the so-called 'subsetting' idea in UML; B: hierarchy of property characteristics relevant for OWL 2.

Outline Sub-Property compatibility Service (*SubProS*)

- First part extends the basic notions from the *RBox compatibility* [Keet and Artale(2008)] (defined for $\mathcal{ALCQI}$)
- Informally, it first checks the 'compatibility' of domain and range axioms w.r.t the object property hierarchy and the class hierarchy.

Outline Sub-Property compatibility Service (*SubProS*)

- First part extends the basic notions from the *RBox compatibility* [Keet and Artale(2008)] (defined for $\mathcal{ALCQI}$)
- Informally, it first checks the 'compatibility' of domain and range axioms w.r.t the object property hierarchy and the class hierarchy.
- After that, *SubProS* checks whether the object property characteristic(s) conform to specification, provided there is such an expression in the ontology.
- It exhaustively checks each permutation of domain and range and then of the characteristic of the parent and child property in the object property hierarchy

Definition (Sub-Property compatibility Service (*SubProS*))

For each pair of object properties, $R, S \in \mathcal{O}$ such that $\mathcal{O} \models S \sqsubseteq R$, and $\mathcal{O}$ an OWL ontology adhering to the syntax and semantics as specified in OWL 2 Standard, check whether:

- Test 1. $\mathcal{O} \models D_S \sqsubseteq D_R$ and $\mathcal{O} \models R_S \sqsubseteq R_R$;
- Test 2. $\mathcal{O} \not\models D_R \sqsubseteq D_S$;
- Test 3. $\mathcal{O} \not\models R_R \sqsubseteq R_S$;
- Test 4. If $\mathcal{O} \models \text{Asym}(R)$ then $\mathcal{O} \models \text{Asym}(S)$;
- Test 5. If $\mathcal{O} \models \text{Sym}(R)$ then $\mathcal{O} \models \text{Sym}(S)$ or $\mathcal{O} \models \text{Asym}(S)$;
- Test 6. If $\mathcal{O} \models \text{Trans}(R)$ then $\mathcal{O} \models \text{Trans}(S)$;
- Test 7. If $\mathcal{O} \models \text{Ref}(R)$ then $\mathcal{O} \models \text{Ref}(S)$ or $\mathcal{O} \models \text{Irr}(S)$;
- Test 8. If $\mathcal{O} \models \text{Irr}(R)$ then $\mathcal{O} \models \text{Irr}(S)$ or $\mathcal{O} \models \text{Asym}(S)$;
- Test 9. If $\mathcal{O} \models \text{Asym}(R)$ then $\mathcal{O} \not\models \text{Sym}(S)$;
- Test 10. If $\mathcal{O} \models \text{Irr}(R)$ then $\mathcal{O} \not\models \text{Ref}(S)$;

continues....

Definition (Sub-Property compatibility Service (*SubProS*))

... *continued from previous page*

Test 10. If $\mathcal{O} \models \mathsf{Irr}(R)$ then $\mathcal{O} \not\models \mathsf{Ref}(S)$;

Test 11. If $\mathcal{O} \models \mathsf{Trans}(R)$ then $\mathcal{O} \not\models \mathsf{Irr}(R)$, $\mathcal{O} \not\models \mathsf{Asym}(R)$, $\mathcal{O} \not\models \mathsf{Irr}(S)$, and $\mathcal{O} \not\models \mathsf{Asym}(S)$;

An OWL object property hierarchy is said to be compatible iff

- *Test 1* and (*2* or *3*) hold for all pairs of property-subproperty in $\mathcal{O}$, and
- *Tests 4–11* hold for all pairs of property-subproperty in $\mathcal{O}$.

Property chains

- Recall the three cases for property chains, with $w \sqsubseteq R$:
  - Case S: $w = S_1 \circ \ldots \circ S_n$ and $S_i \prec R$, for all $1 \geq i \geq n$, or
  - Case RS: $w = R \circ S_1 \circ \ldots \circ S_n$ and $S_i \prec R$, for all $1 \geq i \geq n$, or
  - Case SR: $w = S_1 \circ \ldots \circ S_n \circ R$ and $S_i \prec R$, for all $1 \geq i \geq n$.

Property chains

- Recall the three cases for property chains, with $w \sqsubseteq R$:
  - Case S: $w = S_1 \circ \ldots \circ S_n$ and $S_i \prec R$, for all $1 \geq i \geq n$, or
  - Case RS: $w = R \circ S_1 \circ \ldots \circ S_n$ and $S_i \prec R$, for all $1 \geq i \geq n$, or
  - Case SR: $w = S_1 \circ \ldots \circ S_n \circ R$ and $S_i \prec R$, for all $1 \geq i \geq n$.
- To ensure avoidance of undesirable classifications or inconsistencies, informally:
  - The domain/range class from left to right has to be equal or a superclass, on the lhs of the inclusion
  - Similarly for the outer domain and range on the lhs and domain and range of the object property on the rhs

Definition (Property Chain Compatibility Service (*ProChainS*))

For each set of object properties, $R, S_1, \ldots, S_n \in \mathcal{R}$, $\mathcal{R}$ the set of OWL object properties (V_{OP} in OWL 2) in OWL ontology $\mathcal{O}$, and $S_i \prec R$ with $1 \leq i \leq n$, $\mathcal{O}$ adheres to the constraints of Definition 3 (and, more generally, the OWL 2 specification), and user-defined domain and range axioms as defined in Definition 1, for each of the property chain expression, select either one of the three cases:

Case S. Property chain pattern as $S_1 \circ S_2 \circ \ldots \circ S_n \sqsubseteq R$. Test whether:

Test S-a. $\mathcal{O} \models R_{S1} \sqsubseteq D_{S2}, \ldots, R_{Sn-1} \sqsubseteq D_{Sn}$;
Test S-b. $\mathcal{O} \models D_{S1} \sqsubseteq D_R$;
Test S-c. $\mathcal{O} \models R_{Sn} \sqsubseteq R_R$;

Case RS. Property chain pattern as $R \circ S_1 \circ \ldots \circ S_n \sqsubseteq R$. Test whether:

Test RS-a. $\mathcal{O} \models R_{S1} \sqsubseteq D_{S2}, \ldots, R_{Sn-1} \sqsubseteq D_{Sn}$;
Test RS-b. $\mathcal{O} \models R_R \sqsubseteq D_{S1}$;
Test RS-c. $\mathcal{O} \models R_{Sn} \sqsubseteq R_R$;

continues...

Case SR. Property chain pattern as $S_1 \circ \ldots \circ S_n \circ R \sqsubseteq R$. Test

Definition (Property Chain Compatibility Service (*ProChainS*))

.... *continued from previous page*

Case SR. Property chain pattern as $S_1 \circ \ldots \circ S_n \circ R \sqsubseteq R$. Test whether:

Test SR-a. $\mathcal{O} \models R_{S1} \sqsubseteq D_{S2}, \ldots, R_{Sn-1} \sqsubseteq D_{Sn}$;
Test SR-b. $\mathcal{O} \models D_{S1} \sqsubseteq D_R$;
Test SR-c. $\mathcal{O} \models R_{Sn} \sqsubseteq D_R$;

An OWL property chain expression is said to be compatible iff the OWL 2 syntactic constraints hold and either `Case S`, or `Case RS`, or `Case SR` holds.

BioTop's inconsistent 'has process role'

'has process role' in BioTop [Beisswanger et al.(2008)] (v. June 17, 2010) is inconsistent. Relevant axioms are:

$$\text{'has process role'} \sqsubseteq \text{'temporally related to'} \tag{E.1}$$

$$\text{'has process role'} \sqsubseteq \text{'processual entity'} \times \text{role} \tag{E.2}$$

$$\text{'temporally related to'} \sqsubseteq \text{'processual entity'} \sqcup \text{quality} \times \text{'processual entity'} \sqcup \text{quality} \tag{E.3}$$

$$\text{role} \sqsubseteq \neg\text{quality} \tag{E.4}$$

$$\text{role} \sqsubseteq \neg\text{'processual entity'} \tag{E.5}$$

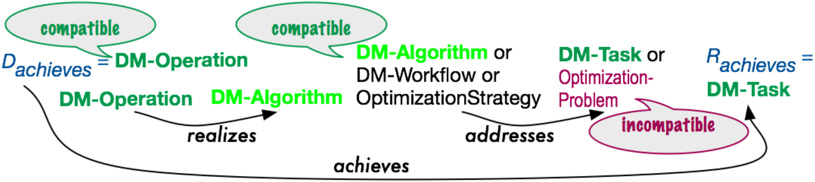
$$\text{Sym}(\text{'temporally related to'}) \tag{E.6}$$

BioTop's inconsistent 'has process role'

Use *SubProS* to isolate the flaw:

- `Test 1`: fail, because $R_{\text{hasprocessrole}} \sqsubseteq R_{\text{temporallyrelatedto}}$ is false, as the ranges (see E.2 cf. E.3) are disjoint (see E.4, E.5) and therewith 'has process role' is inconsistent;
- `Test 2` and 3: pass.
- `Test 4`: not applicable.
- `Test 5`: fail, because $\mathcal{O}$ does not contain Sym('has process role').
- `Test 6–11`: not applicable.

DMOP chain in v5.2



Of type *Case S*. `Test S-c` (for corrections) failed because $\mathcal{O} \not\models R_{\text{DM-Task}\sqcup\text{OptimizationProblem}} \sqsubseteq R_{\text{DM-Task}}$. Considering the suggestions for revision, step B's first option to revise the ontology was chosen, i.e., removing OptimizationProblem from the range axiom of addresses.

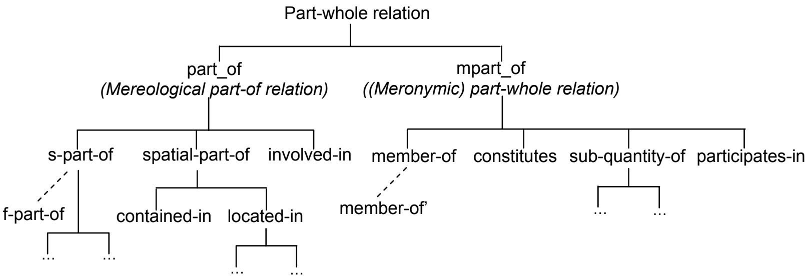
Don't reinvent the wheel

- Part-whole relations, probably received most attention in ontologies
- Spatial relations, and its interaction with parthood
- Participation, constitution, causation, ...
- Similarity: important for combination machine learning with ontologies

Taxonomy of part-whole relations

- Hierarchy of part-whole relations common in ontologies and conceptual data models
- Uses DOLCE foundational ontology [Masolo et al.(2003)] for domain and range of a relation
- Main distinction between transitive (parthood) vs non-transitive (just meronymic) part-whole relations
- Formally defined
- Details in [Keet and Artale(2008)]

Part-whole relations



Part-whole relations

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

$$\forall x, y(member\_of_n(x, y) \triangleq mpart\_of(x, y) \land (POB(x) \lor SOB(x)) \land SOB(y))$$

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

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

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`)

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

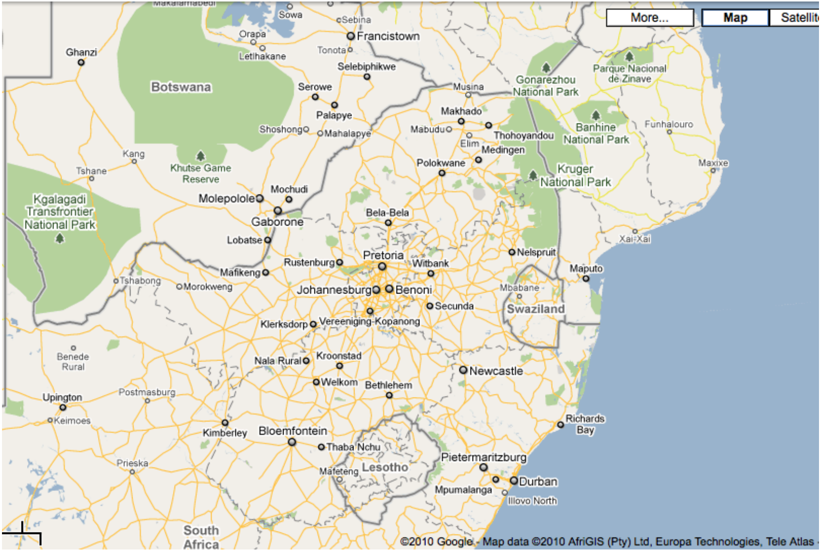
Object and its 2D or 3D region, such as `contained_in(John's address book, John's bag)` and `located_in(Pretoria, South Africa)`

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

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

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

Knowledge and Google & AfriGIS



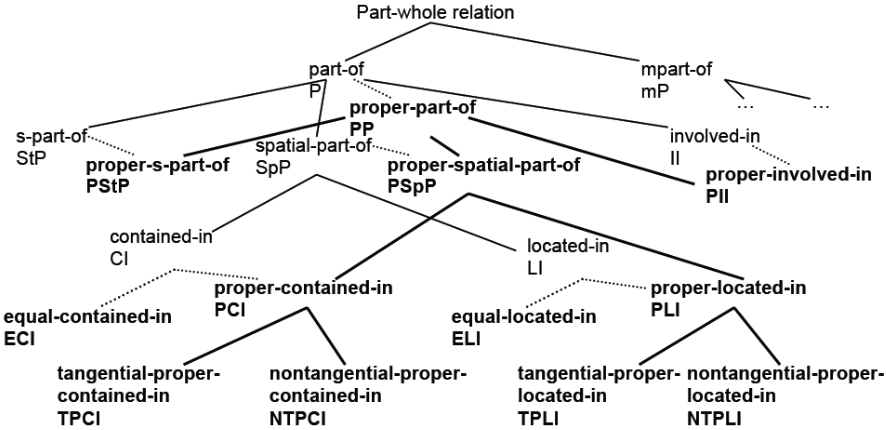
Parts and space

- Could not represent all of parthood in OWL or any DL, worse for mereotopology, but tried anyway [Keet et al.(2012)]
- Example:
  - Let `NTPLI` be a 'non-tangential proper located in' relation
  - $\texttt{EnclosedCountry} \equiv \texttt{Country} \sqcap \exists\texttt{NTPLI.Country}$
  - `NTPLI(Lesotho, South Africa)`, `Country(Lesotho)`, `Country(South Africa)`,
  - then it will correctly deduce `EnclosedCountry(Lesotho)`.
  - with merely 'part-of', one would not have been able to obtain this result

Some common relations

- 9-Intersection Method (9IM), based on point-set topology [Egenhofer and Herring(1990)]
- Region Connection Calculus (RCC), based on the reflexive and symmetric *connection* [Randell et al.(1992)]
- Neither one considers the combination of the space region with the object that occupies it
- This interaction is addressed by **mereotopology**, which focuses on spatial *entities*, not just regions.

Integrate the extension [Keet et al.(2012)]



Kuratowski General Extensional MereoTopology

Core axioms and definitions			
$P(x,x)$	(t1)	$P(x,y) \wedge P(y,z) \rightarrow P(x,z)$	(t2)
$P(x,y) \wedge P(y,x) \rightarrow x = y$	(t3)	$\neg P(y,x) \rightarrow \exists z(P(z,y) \wedge \neg O(z,x))$	(t4)
$\exists w \phi(w) \rightarrow \exists z \forall w(O(w,z) \leftrightarrow \exists v(\phi(v) \wedge O(w,v)))$			(t5)
$C(x,x)$	(t6)	$C(x,y) \rightarrow C(y,x)$	(t7)
$P(x,y) \rightarrow E(x,y)$	(t8)	$E(x,y) =_{df} \forall z(C(z,x) \rightarrow C(z,y))$	(t9)
$E(x,y) \rightarrow P(x,y)$	(t10)	$SC(x) \leftrightarrow \forall y,z(x = y + z \rightarrow C(y,z))$	(t11)
$\exists z(SC(z) \wedge O(z,x) \wedge O(z,y) \wedge \forall w(P(w,z) \rightarrow (O(w,x) \vee O(w,y)))) \rightarrow C(x,y)$			(t12)
$z = \sum x \phi x \rightarrow \forall y(C(y,z) \rightarrow \exists x(\phi x \wedge C(y,x)))$			(t13)
$P(x,cx)$	(t14)	$c(cx) = cx$	(t15)
$c(x+y) = cx + cy$	(t16)	$cx =_{df} \sim(ex)$	(t17)
$ex =_{df} i(\sim x)$	(t18)	$ix =_{df} \sum z \forall y(C(z,y) \rightarrow O(x,y))$	(t19)
Additional axioms, definitions, and theorems			
$PP(x,y) =_{df} P(x,y) \wedge \neg P(y,x)$	(t20)	$O(x,y) =_{df} \exists z(P(z,x) \wedge P(z,y))$	(t21)
$EQ(x,y) =_{df} P(x,y) \wedge P(y,x)$	(t22)	$TPP(x,y) =_{df} PP(x,y) \wedge \neg IPP(x,y)$	(t23)
$IPP(x,y) =_{df} PP(x,y) \wedge \forall z(C(z,x) \rightarrow O(z,y))$			(t24)
$\neg PP(x,x)$	(t25)	$PP(x,y) \wedge PP(y,z) \rightarrow PP(x,z)$	(t26)
$PP(x,y) \rightarrow \neg PP(y,x)$	(t27)		

details

Subsets of KGEMT that can be represented in OWL

- Reason of differences: the object property characteristics (e.g. t1/t6 = ref. of P/C, t25 = irr. of PP, t2= trans.).
- The six definitions (PP, O, TPP, etc.) can be simplified and added as primitives to each one.

OWL species	Subsets of KGEMT axioms
OWL 2 DL	(t1, t2, t6, t7, t8, t10, t26) or (t1, t2, t6, t7, t8, t10, t27) or (t1, t2, t6, t7, t8, t10, t25)
OWL DL	t2, t7, t8, t10, t26
OWL Lite	t2, t7, t8, t10, t26
OWL 2 RL	t2, t7, t8, t10, t26
OWL 2 EL	t1, t2, t6, t8, t10, t26
OWL 2 QL	t1, t6, t7, t8, t10

- Importance depends on the desired inference scenarios; thus far, Trans, Sym, Asym, and Irr seem to be more interesting, i.e., giving precedence to OWL 2 DL and OWL 2 RL (See [Keet et al.(2012)] for details on reasoning trade-offs)

Other relations in (foundational) ontologies

- Relation Ontology [Smith et al.(2005)]
- Relations that are sort-of temporal, but now not used as such; hence, one cannot reason ‘fully’ with them w.r.t. intended meaning
  - e.g.: `derived-from`, `transformation-of`
- dependence, inherence
- Attributes
- DOLCE’s qualities

Some other aspects of relations (not covered now)

- constraints on participation (essential vs. immutable vs. mandatory)
- Modality, necessity, telic, atelic
- Temporal relations, relation migration
- *n*-ary relations and reifying (objectifying) them

Any suggestions for actual ontology development?

- Using the taxonomy of part-whole relations
- Reasoner-guided relation selection
- Performance tradeoffs with inverses

Using the taxonomy of part-whole relations

- Representing it correctly in ontologies and conceptual data models

- Reasoning with a taxonomy of relations

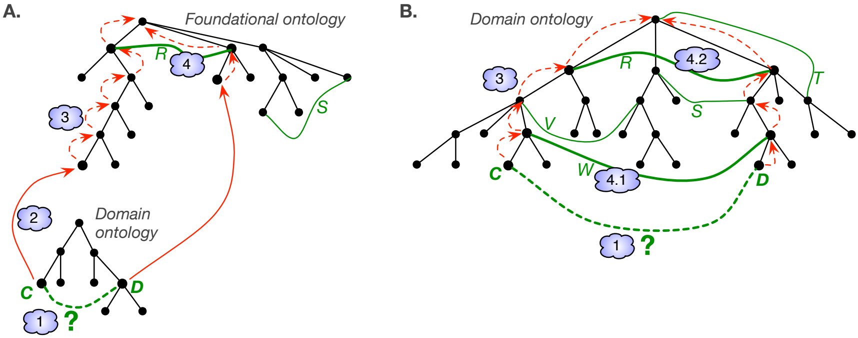
Using the taxonomy of part-whole relations

- 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: ONTOPARTS [Keet et al.(2012)] and OntoParts-2 [Keet et al.(2013b)]
- Reasoning with a taxonomy of relations

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: ONTOPARTS [Keet et al.(2012)] and OntoParts-2 [Keet et al.(2013b)]
- Reasoning with a taxonomy of relations
  - The *RBox reasoning service* [Keet and Artale(2008)] or *SubProS* [Keet(2014)] to pinpoint errors

GENERATOR: Guided ENtity reuse and class Expression geneRATOR



[Keet et al.(2013a)]

Effects of features on reasoning

- Disjoint OPs, reflexivity, and qualified cardinality only on *simple* OPs in OWL 2. with *non-simple* when:
  - if $\mathcal{O}$ contains an axiom $S \circ T \sqsubseteq R$
  - if R is non-simple, then so is its inverse R^-
  - if R is non-simple and $\mathcal{O}$ contains any of the axioms $R \sqsubseteq S$, $S \equiv R$ or $R \equiv S$, then S is also non-simple
- Domain and range axioms
- Role hierarchy with domain and range axioms vs. 'specialising' in class axioms (with existential) [Hammar(2014)]
- Inverses
- 'understanding' the reasoner, predicting performance a hot topic; e.g. [Goncalves et al.(2012), Kang et al.(2012)]

Outline

Summary

1. Methodologies
   - Macro-level methodologies
   - Micro-level methodologies
   - Test-Driven Development
2. Modelling
   - Relationship issues
   - Semantics of relations
   - Some common relations
   - Reasoner-mediated modelling
3. Summary
   - Exercises

References I

S. Auer.
The RapidOWL methodology–towards Agile knowledge engineering.
In *Proc. of WETICE'06*, pages 352–357. IEEE Computer Society, June 2006.
doi: 10.1109/WETICE.2006.67.

Kent Beck.
Test-Driven Development: by example.
Addison-Wesley, Boston, MA, 2004.

Elena Beisswanger, Stefan Schulz, Holger Stenzhorn, and Udo Hahn.
BioTop: An upper domain ontology for the life sciences - a description of its current structure, contents, and interfaces to OBO ontologies.
Applied Ontology, 3(4):205–212, 2008.

E Blomqvist, A. S. Sepour, and V. Presutti.
Ontology testing – methodology and tool.
In *Proc. of EKAW'12*, volume 7603 of *LNAI*, pages 216–226. Springer, 2012.

Kieren Davies.
Towards test-driven development of ontologies: An analysis of testing algorithms.
Project report, University of Cape Town, 2016.
https://people.cs.uct.ac.za/~dvskie001/doc/TDD_Ontologies_Analysis_of_Testing_Algorithms.pdf.

References II

Kieren Davies, C. Maria Keet, and Agnieszka Lawrynowicz.
Tddonto2: A test-driven development plugin for arbitrary TBox and ABox axioms.
In *Proc. of ESWC'17 Posters and Demos*, LNCS, page in print. Springer, 2017.
30 May - 1 June 2017, Portoroz, Slovenia.

M. J. Egenhofer and J. R. Herring.
Categorizing binary topological relations between regions, lines, and points in geographic databases.
Technical Report 90-12, National Center for Geographic Information and Analysis, University of California, 1990.

S. Ferré.
Semantic authoring of ontologies by exploration and elimination of possible worlds.
In E. Blomqvist, P. Ciancarini, F. Poggi, and F. Vitali, editors, *Proceedings of the 20th International Conference on Knowledge Engineering and Knowledge Management (EKAW'16)*, volume 10024 of *LNAI*, pages 180–195. Springer, 2016.
19-23 November 2016, Bologna, Italy.

Sebastien Ferré and Sebastian Rudolph.
Advocatus diaboli exploratory enrichment of ontologies with negative constraints.
In *Proc. of EKAW'12*, volume 7603 of *LNAI*, pages 42–56. Springer, 2012.
Oct 8-12, Galway, Ireland.

Pablo Rubén Fillottrani and C. Maria Keet.
Evidence-based languages for conceptual data modelling profiles.
In T. Morzy et al., editors, *Proceedings of the 19th Conference on Advances in Databases and Information Systems (ADBIS'15)*, volume 9282 of *LNCS*, pages 215–229. Springer, 2015.
8-11 Sept, 2015, Poitiers, France.

References III

Kit Fine.
Neutral relations.
The Philosophical Review, 109(1):1–33, 2000.

S. Garca-Ramos, A. Otero, and M Fernández-López.
OntologyTest: A tool to evaluate ontologies through tests defined by the user.
In *Proc. of IWANN 2009 Workshops, Part II*, volume 5518 of *LNCS*, pages 91–98. Springer, 2009.

Alexander Garcia, Kieran O'Neill, Leyla Jael Garcia, Phillip Lord, Robert Stevens, Óscar Corcho, and Frank Gibson.
Developing ontologies within decentralized settings.
In H. Chen et al., editors, *Semantic e-Science. Annals of Information Systems 11*, pages 99–139. Springer, 2010.

R.S. Goncalves, B. Parsia, and U. Sattler.
Performance heterogeneity and approximate reasoning in description logic ontologies.
In *Proceedings of the International Semantic Web Conference 2012*, LNCS, pages 82–98. Springer, 2012.

N. Guarino and G. Guizzardi.
We need to discuss the relationship: Revisiting relationships as modeling constructs.
In Jelena Zdravkovic, Marite Kirikova, and Paul Johannesson, editors, *Proceedings of the 27th International Conference on Advance Information Systems Engineering (CAISE'15)*, volume 9097 of *LNCS*, pages 279–294, 2015.

References IV

N. Guarino and G. Guizzardi.
Relationships and events: Towards a general theory of reification and truthmaking.
In Giovanni Adorni, Stefano Cagnoni, Marco Gori, and Marco Maratea, editors, *Proceedings of the 15th International Conference of the Italian Association for Artificial Intelligence (AI*IA 2016)*, volume 10037 of *LNAI*, pages 237–249. Springer, 2016.

Giancarlo Guizzardi and Gerd Wagner.
What's in a relationship: An ontological analysis.
In Qing Li, Stefano Spaccapietra, Eric Yu, and Antoni Olivé, editors, *ER*, volume 5231 of *Lecture Notes in Computer Science*, pages 83–97. Springer, 2008.
ISBN 978-3-540-87876-6.

K. Hammar.
Ontology design pattern property specialisation strategies.
In K. Janowicz and S. Schlobach, editors, *19th International Conference on Knowledge Engineering and Knowledge Management (EKAW'14)*, volume 8876 of *LNAI*, pages 165–180. Springer, 2014.
24-28 Nov, 2014, Linkoping, Sweden.

M. Horridge, B. Parsia, and U. Sattler.
Laconic and precise justifications in OWL.
In *Proc. of the 7th International Semantic Web Conference (ISWC 2008)*, volume 5318 of *LNCS*. Springer, 2008.

I. Horrocks, O. Kutz, and U. Sattler.
The even more irresistible $\mathcal{SROIQ}$.
Proceedings of KR-2006, pages 452–457, 2006.

References V

David S. Janzen.
Software architecture improvement through test-driven development.
In *Companion to ACM SIGPLAN'05*, pages 240–241. ACM Proceedings, 2005.

A. Kalyanpur, B. Parsia, E. Sirin, and B. Grau.
Repairing unsatisfiable concepts in OWL ontologies.
In Y Sure and J. Domingue, editors, *Proceedings of the European Semantic Web Conference (ESWC'06)*, volume 4011 of *LNCS*. Springer, 2006.

Y.-B. Kang, Y.-F. Li, and S. Krishnaswamy.
Predicting reasoning performance using ontology metrics.
In *Proceedings of the International Semantic Web Conference 2012*, LNCS, pages 198–214. Springer, 2012.

C. M. Keet and T. Chirema.
A model for verbalising relations with roles in multiple languages.
In E. Blomqvist, P. Ciancarini, F. Poggi, and F. Vitali, editors, *Proceedings of the 20th International Conference on Knowledge Engineering and Knowledge Management (EKAW'16)*, volume 10024 of *LNAI*, pages 384–399. Springer, 2016.
19-23 November 2016, Bologna, Italy.

C. M. Keet and A. Ławrynowicz.
Test-driven development of ontologies.
In *13th Extended Semantic Web Conference (ESWC'16)*, LNCS. Springer, 2016.
29 May - 2 June, 2016, Crete, Greece.

References VI

C. M. Keet, M. C. Suárez-Figueroa, and M. Poveda-Villalón.
Pitfalls in ontologies and tips to prevent them.
In A. Fred, J. L. G. Dietz, K. Liu, and J. Filipe, editors, *Knowledge Discovery, Knowledge Engineering and Knowledge Management: IC3K 2013 Selected papers*, volume 454 of *CCIS*, pages 115–131. Springer, 2015a.

C. Maria Keet.
Positionalism of relations and its consequences for fact-oriented modelling.
In R. Meersman, P. Herrero, and Dillon T., editors, *OTM Workshops, International Workshop on Fact-Oriented Modeling (ORM'09)*, volume 5872 of *LNCS*, pages 735–744. Springer, 2009.
Vilamoura, Portugal, November 4-6, 2009.

C. Maria Keet.
Detecting and revising flaws in OWL object property expressions.
In A. ten Teije et al., editors, *18th International Conference on Knowledge Engineering and Knowledge Management (EKAW'12)*, volume 7603 of *LNAI*, pages 252–266. Springer, 2012.
URL `http://www.meteck.org/files/EKAW12subProsChains.pdf`.
Oct 8-12, Galway, Ireland.

C. Maria Keet.
Preventing, detecting, and revising flaws in object property expressions.
Journal on Data Semantics, 3(3):189–206, 2014.

C. Maria Keet and Alessandro Artale.
Representing and reasoning over a taxonomy of part-whole relations.
Applied Ontology – Special issue on Ontological Foundations for Conceptual Modeling, 3(1-2):91–110, 2008.

References VII

C. Maria Keet and Pablo Rubén Fillottrani.
An ontology-driven unifying metamodel of UML Class Diagrams, EER, and ORM2.
Data & Knowledge Engineering, 98:30–53, 2015.
doi: 0.1016/j.datak.2015.07.004.

C. Maria Keet, Francis C. Fernández-Reyes, and Annette Morales-González.
Representing mereotopological relations in OWL ontologies with ONTOPARTS.
In E. Simperl et al., editors, *Proc.of ESWC'12*, volume 7295 of *LNCS*, pages 240–254. Springer, 2012.
29-31 May 2012, Heraklion, Crete, Greece.

C. Maria Keet, M. Tahir Khan, and Chiara Ghidini.
Guided ENtity reuse and class Expression geneRATOR.
In Richard Benjamins, Mathieu d'Aquin, Andrew Gordon, , and José Manuel Gómez-Pérez, editors, *Seventh International Conference on Knowledge Capture (K-CAP'13)*, page a26 (poster&demo). ACM, 2013a.
23-26 June 2013, Banff, Canada.

C. Maria Keet, M. Tahir Khan, and Chiara Ghidini.
Ontology authoring with FORZA.
In *Proceedings of the 22nd ACM international conference on Conference on Information & Knowledge Management (CIKM'13)*, pages 569–578. ACM proceedings, 2013b.
Oct. 27 - Nov. 1, 2013, San Francisco, USA.

C. Maria Keet, Agnieszka Lawrynowicz, Claudia d'Amato, Alexandros Kalousis, P. Nguyen, Raul Palma, Robert Stevens, and Melani Hilario.
The data mining optimization ontology.
Web Semantics: Science, Services and Agents on the World Wide Web, 32:43–53, 2015b.
doi: 10.1016/j.websem.2015.01.001.

References VIII

Ilianna Kollia, Birte Glimm, and Ian Horrocks.
SPARQL Query Answering over OWL Ontologies.
In *Proc, of ESWC'11*, volume 6643 of *LNCS*, pages 382–396. Springer, 2011.
29 May-2 June, 2011, Heraklion, Crete, Greece.

D. Kontokostas, P. Westphal, Sören Auer, Sebastian Hellmann, Jens Lehmann, Roland Cornelissen, and Amrapali Zaveri.
Test-driven evaluation of linked data quality.
In *Proc. of WWW'14*, pages 747–758. ACM proceedings, 2014.

Shaweta Kumar and Sanjeev Bansal.
Comparative study of test driven development with traditional techniques.
Int. J. Soft Comp. & Eng., 3(1):352–360, 2013.

A. Ławrynowicz and C. M. Keet.
The tddonto tool for test-driven development of dl knowledge bases.
In R. Peñaloza and M. Lenzerini, editors, *29th International Workshop on Description Logics (DL'16)*, volume 1577 of *CEUR-WS*, 2016.
22-25 April 2016, Cape Town, South Africa.

Joop Leo.
Modeling relations.
Journal of Philosophical Logic, 37:353–385, 2008.

References IX

C. Masolo, S. Borgo, A. Gangemi, N. Guarino, and A. Oltramari.
Ontology library.
WonderWeb Deliverable D18 (ver. 1.0, 31-12-2003)., 2003.
http://wonderweb.semanticweb.org.

B. Parsia, E. Sirin, and A. Kalyanpur.
Debugging OWL ontologies.
In *Proceedings of the World Wide Web Conference (WWW 2005)*, 2005.
May 10-14, 2005, Chiba, Japan.

Adrian Paschke and Ralph Schaefermeier.
Aspect OntoMaven - aspect-oriented ontology development and configuration with OntoMaven.
Technical Report 1507.00212v1, Free University of Berlin, July 2015.
URL `http://arxiv.org/abs/1507.00212`.

María Poveda-Villalón, Mari Carmen Suárez-Figueroa, and Asunción Gómez-Pérez.
Validating ontologies with OOPS!
In A. ten Teije et al., editors, *18th International Conference on Knowledge Engineering and Knowledge Management (EKAW'12)*, volume 7603 of *LNAI*, pages 267–281. Springer, 2012.
Oct 8-12, Galway, Ireland.

V. Presutti, E Daga, et al.
extreme design with content ontology design patterns.
In *Proc. of WS on OP'09*, volume 516 of *CEUR-WS*, pages 83–97, 2009.

References X

V. Presutti et al.
A library of ontology design patterns: reusable solutions for collaborative design of networked ontologies.
NeOn deliverable D2.5.1, NeOn Project, ISTC-CNR, 2008.

D. A. Randell, Z. Cui, and A. G. Cohn.
A spatial logic based on regions and connection.
In *Proc. 3rd Int. Conf. on Knowledge Representation and Reasoning*, pages 165–176. Morgan Kaufmann, 1992.

Yuan Ren, Artemis Parvizi, Chris Mellish, Jeff Z. Pan, Kees van Deemter, and Robert Stevens.
Towards competency question-driven ontology authoring.
In *Proc. of ESWC'14*, volume 8465 of *LNCS*, page 752767. Springer, 2014.

B. Smith, W. Ceusters, B. Klagges, J. Köhler, A. Kumar, J. Lomax, C. Mungall, F. Neuhaus, A. L. Rector, and C. Rosse.
Relations in biomedical ontologies.
Genome Biology, 6:R46, 2005.

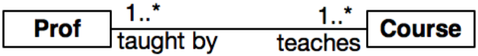
Danny Vrandečić and Aldo Gangemi.
Unit tests for ontologies.
In *OTM workshops 2006*, volume 4278 of *LNCS*, pages 1012–1020. Springer, 2006.

J. D. Warrender and P. Lord.
How, What and Why to test an ontology.
Technical Report 1505.04112, Newcastle University, 2015.
http://arxiv.org/abs/1505.04112.

Choose one involvement between Chewing and Eating

- Chewing involved-in some Eating
  $\texttt{Chewing} \sqsubseteq \exists \texttt{involved-in.Eating}$
- Chewing inverse(involves) some Eating
  $\texttt{Chewing} \sqsubseteq \exists \texttt{involves}^{-}.\texttt{Eating}$
- Eating involves some Chewing
  $\texttt{Eating} \sqsubseteq \exists \texttt{involves.Chewing}$
- Eating inverse(involved-in) some Chewing
  $\texttt{Eating} \sqsubseteq \exists \texttt{involved-in}^{-}.\texttt{Chewing}$

How to formalise the UML diagram in OWL?



- teaches, taught-by, InverseObjectProperties(teaches taught-by)
  $\texttt{teaches} \sqsubseteq \top \times \top$
  $\texttt{taughtBy} \sqsubseteq \top \times \top$
  $\texttt{teaches} \equiv \texttt{taughtBy}^{-}$
- domain teaches: Prof, and range teaches: Course
  $\texttt{teaches} \sqsubseteq \texttt{Prof} \times \texttt{Course}$
- domain teaches: Prof, and range teaches: Course, domain taught-by: Course, range taught-by: Prof
  $\texttt{teaches} \sqsubseteq \texttt{Prof} \times \texttt{Course}$
  $\texttt{taughtBy} \sqsubseteq \texttt{Course} \times \texttt{Prof}$

OWL files

- http://www.meteck.org/teaching/ontologies/ has various versions of the African Wildlife Ontology (alone, linked to DOLCE, link to GFO)
- http://www.meteck.org/files/ontologies/EvalComputer.owl has no object properties at all. add both properties and axioms (details of exercise depends on number of participants)
- Pick one. Add missing object properties and/or axioms (details of exercise depends on number of participants)

The Wildlife Ontology and DOLCE

- Giraffes eat leaves and twigs. how do `Plant` and `Twig` relate?
    -
- The elephant's tusks (ivory) are made of apatite (calcium phosphate); which DOLCE relation can be reused?
    -
- How would you represent the `Size` (`Height`, `Weight`, etc.) of an average adult elephant?
    - with *quality* and *quale*
    - OWL data properties
        -
        -
        -

The Wildlife Ontology and DOLCE

- Giraffes eat leaves and twigs. how do `Plant` and `Twig` relate?
  - (some type of) parthood relation
- The elephant's tusks (ivory) are made of apatite (calcium phosphate); which DOLCE relation can be reused?
  - constitution
- How would you represent the `Size` (`Height`, `Weight`, etc.) of an average adult elephant?
  - with *quality* and *quale*
  - OWL data properties

The Wildlife Ontology and DOLCE

- Giraffes eat leaves and twigs. how do `Plant` and `Twig` relate?
    - (some type of) parthood relation
- The elephant's tusks (ivory) are made of apatite (calcium phosphate); which DOLCE relation can be reused?
    - constitution
- How would you represent the `Size` (`Height`, `Weight`, etc.) of an average adult elephant?
    - with *quality* and *quale*
    - OWL data properties
        - What is the data type; integer, float, real, string?
        - Measure in meter, feet, kg, lb?
        - Introduce "`ElephantHeight`", and also "`LionHeight`", "`GiraffeHeight`', "`ImpalaHeight`", etc.?

A computer ontology

- CPU and Desktop?
- Who are members of an Agile team?

A computer ontology

- CPU and Desktop?
  - containment
- Who are members of an Agile team?
  - hasMember vs. memberOf

Ground Topology

Core axioms and definitions			
$P(x,x)$	(t1)	$P(x,y) \wedge P(y,z) \rightarrow P(x,z)$	(t2)
$P(x,y) \wedge P(y,x) \rightarrow x = y$	(t3)	$\neg P(y,x) \rightarrow \exists z(P(z,y) \wedge \neg O(z,x))$	(t4)
$\exists w \phi(w) \rightarrow \exists z \forall w(O(w,z) \leftrightarrow \exists v(\phi(v) \wedge O(w,v)))$			(t5)
$C(x,x)$	(t6)	$C(x,y) \rightarrow C(y,x)$	(t7)
$P(x,y) \rightarrow E(x,y)$	(t8)	$E(x,y) =_{df} \forall z(C(z,x) \rightarrow C(z,y))$	(t9)
$E(x,y) \rightarrow P(x,y)$	(t10)	$SC(x) \leftrightarrow \forall y,z(x = y + z \rightarrow C(y,z))$	(t11)
$\exists z(SC(z) \wedge O(z,x) \wedge O(z,y) \wedge \forall w(P(w,z) \rightarrow (O(w,x) \vee O(w,y)))) \rightarrow C(x,y)$			(t12)
$z = \sum x \phi x \rightarrow \forall y(C(y,z) \rightarrow \exists x(\phi x \wedge C(y,x)))$			(t13)
$P(x,cx)$	(t14)	$c(cx) = cx$	(t15)
$c(x+y) = cx + cy$	(t16)	$cx =_{df} \sim (ex)$	(t17)
$ex =_{df} i(\sim x)$	(t18)	$ix =_{df} \sum z \forall y(C(z,y) \rightarrow O(x,y))$	(t19)
Additional axioms, definitions, and theorems			
$PP(x,y) =_{df} P(x,y) \wedge \neg P(y,x)$	(t20)	$O(x,y) =_{df} \exists z(P(z,x) \wedge P(z,y))$	(t21)
$EQ(x,y) =_{df} P(x,y) \wedge P(y,x)$	(t22)	$TPP(x,y) =_{df} PP(x,y) \wedge \neg IPP(x,y)$	(t23)
$IPP(x,y) =_{df} PP(x,y) \wedge \forall z(C(z,x) \rightarrow O(z,y))$			(t24)
$\neg PP(x,x)$	(t25)	$PP(x,y) \wedge PP(y,z) \rightarrow PP(x,z)$	(t26)
$PP(x,y) \rightarrow \neg PP(y,x)$	(t27)		

Minimal (mereo) Topology

Core axioms and definitions			
$P(x,x)$	(t1)	$P(x,y) \wedge P(y,z) \rightarrow P(x,z)$	(t2)
$P(x,y) \wedge P(y,x) \rightarrow x = y$	(t3)	$\neg P(y,x) \rightarrow \exists z(P(z,y) \wedge \neg O(z,x))$	(t4)
$\exists w \phi(w) \rightarrow \exists z \forall w(O(w,z) \leftrightarrow \exists v(\phi(v) \wedge O(w,v)))$			(t5)
$C(x,x)$	(t6)	$C(x,y) \rightarrow C(y,x)$	(t7)
$P(x,y) \rightarrow E(x,y)$	(t8)	$E(x,y) =_{df} \forall z(C(z,x) \rightarrow C(z,y))$	(t9)
$E(x,y) \rightarrow P(x,y)$	(t10)	$SC(x) \leftrightarrow \forall y,z(x = y+z \rightarrow C(y,z))$	(t11)
$\exists z(SC(z) \wedge O(z,x) \wedge O(z,y) \wedge \forall w(P(w,z) \rightarrow (O(w,x) \vee O(w,y)))) \rightarrow C(x,y)$			(t12)
$z = \sum x\phi x \rightarrow \forall y(C(y,z) \rightarrow \exists x(\phi x \wedge C(y,x)))$			(t13)
$P(x,cx)$	(t14)	$c(cx) = cx$	(t15)
$c(x+y) = cx + cy$	(t16)	$cx =_{df} \sim(ex)$	(t17)
$ex =_{df} i(\sim x)$	(t18)	$ix =_{df} \sum z \forall y(C(z,y) \rightarrow O(x,y))$	(t19)
Additional axioms, definitions, and theorems			
$PP(x,y) =_{df} P(x,y) \wedge \neg P(y,x)$	(t20)	$O(x,y) =_{df} \exists z(P(z,x) \wedge P(z,y))$	(t21)
$EQ(x,y) =_{df} P(x,y) \wedge P(y,x)$	(t22)	$TPP(x,y) =_{df} PP(x,y) \wedge \neg IPP(x,y)$	(t23)
$IPP(x,y) =_{df} PP(x,y) \wedge \forall z(C(z,x) \rightarrow O(z,y))$			(t24)
$\neg PP(x,x)$	(t25)	$PP(x,y) \wedge PP(y,z) \rightarrow PP(x,z)$	(t26)
$PP(x,y) \rightarrow \neg PP(y,x)$	(t27)		

Minimal (mereo) Topology; Ground Mereology

Core axioms and definitions			
$P(x,x)$	(t1)	$P(x,y) \wedge P(y,z) \rightarrow P(x,z)$	(t2)
$P(x,y) \wedge P(y,x) \rightarrow x = y$	(t3)	$\neg P(y,x) \rightarrow \exists z(P(z,y) \wedge \neg O(z,x))$	(t4)
$\exists w \phi(w) \rightarrow \exists z \forall w(O(w,z) \leftrightarrow \exists v(\phi(v) \wedge O(w,v)))$			(t5)
$C(x,x)$	(t6)	$C(x,y) \rightarrow C(y,x)$	(t7)
$P(x,y) \rightarrow E(x,y)$	(t8)	$E(x,y) =_{df} \forall z(C(z,x) \rightarrow C(z,y))$	(t9)
$E(x,y) \rightarrow P(x,y)$	(t10)	$SC(x) \leftrightarrow \forall y,z(x = y + z \rightarrow C(y,z))$	(t11)
$\exists z(SC(z) \wedge O(z,x) \wedge O(z,y) \wedge \forall w(P(w,z) \rightarrow (O(w,x) \vee O(w,y)))) \rightarrow C(x,y)$			(t12)
$z = \sum x \phi x \rightarrow \forall y(C(y,z) \rightarrow \exists x(\phi x \wedge C(y,x)))$			(t13)
$P(x, cx)$	(t14)	$c(cx) = cx$	(t15)
$c(x+y) = cx + cy$	(t16)	$cx =_{df} \sim(ex)$	(t17)
$ex =_{df} i(\sim x)$	(t18)	$ix =_{df} \sum z \forall y(C(z,y) \rightarrow O(x,y))$	(t19)
Additional axioms, definitions, and theorems			
$PP(x,y) =_{df} P(x,y) \wedge \neg P(y,x)$	(t20)	$O(x,y) =_{df} \exists z(P(z,x) \wedge P(z,y))$	(t21)
$EQ(x,y) =_{df} P(x,y) \wedge P(y,x)$	(t22)	$TPP(x,y) =_{df} PP(x,y) \wedge \neg IPP(x,y)$	(t23)
$IPP(x,y) =_{df} PP(x,y) \wedge \forall z(C(z,x) \rightarrow O(z,y))$			(t24)
$\neg PP(x,x)$	(t25)	$PP(x,y) \wedge PP(y,z) \rightarrow PP(x,z)$	(t26)
$PP(x,y) \rightarrow \neg PP(y,x)$	(t27)		

Minimal (mereo) Topology; General Extensional Mereology

Core axioms and definitions			
$P(x,x)$	(t1)	$P(x,y) \wedge P(y,z) \rightarrow P(x,z)$	(t2)
$P(x,y) \wedge P(y,x) \rightarrow x = y$	(t3)	$\neg P(y,x) \rightarrow \exists z(P(z,y) \wedge \neg O(z,x))$	(t4)
$\exists w \phi(w) \rightarrow \exists z \forall w(O(w,z) \leftrightarrow \exists v(\phi(v) \wedge O(w,v)))$			(t5)
$C(x,x)$	(t6)	$C(x,y) \rightarrow C(y,x)$	(t7)
$P(x,y) \rightarrow E(x,y)$	(t8)	$E(x,y) =_{df} \forall z(C(z,x) \rightarrow C(z,y))$	(t9)
$E(x,y) \rightarrow P(x,y)$	(t10)	$SC(x) \leftrightarrow \forall y,z(x = y+z \rightarrow C(y,z))$	(t11)
$\exists z(SC(z) \wedge O(z,x) \wedge O(z,y) \wedge \forall w(P(w,z) \rightarrow (O(w,x) \vee O(w,y)))) \rightarrow C(x,y)$			(t12)
$z = \sum x \phi x \rightarrow \forall y(C(y,z) \rightarrow \exists x(\phi x \wedge C(y,x)))$			(t13)
$P(x, cx)$	(t14)	$c(cx) = cx$	(t15)
$c(x+y) = cx + cy$	(t16)	$cx =_{df} \sim (ex)$	(t17)
$ex =_{df} i(\sim x)$	(t18)	$ix =_{df} \sum z \forall y(C(z,y) \rightarrow O(x,y))$	(t19)
Additional axioms, definitions, and theorems			
$PP(x,y) =_{df} P(x,y) \wedge \neg P(y,x)$	(t20)	$O(x,y) =_{df} \exists z(P(z,x) \wedge P(z,y))$	(t21)
$EQ(x,y) =_{df} P(x,y) \wedge P(y,x)$	(t22)	$TPP(x,y) =_{df} PP(x,y) \wedge \neg IPP(x,y)$	(t23)
$IPP(x,y) =_{df} PP(x,y) \wedge \forall z(C(z,x) \rightarrow O(z,y))$			(t24)
$\neg PP(x,x)$	(t25)	$PP(x,y) \wedge PP(y,z) \rightarrow PP(x,z)$	(t26)
$PP(x,y) \rightarrow \neg PP(y,x)$	(t27)		

General Extensional MereoTopology

Core axioms and definitions			
$P(x,x)$	(t1)	$P(x,y) \wedge P(y,z) \to P(x,z)$	(t2)
$P(x,y) \wedge P(y,x) \to x = y$	(t3)	$\neg P(y,x) \to \exists z(P(z,y) \wedge \neg O(z,x))$	(t4)
$\exists w \phi(w) \to \exists z \forall w (O(w,z) \leftrightarrow \exists v(\phi(v) \wedge O(w,v)))$			(t5)
$C(x,x)$	(t6)	$C(x,y) \to C(y,x)$	(t7)
$P(x,y) \to E(x,y)$	(t8)	$E(x,y) =_{df} \forall z(C(z,x) \to C(z,y))$	(t9)
$E(x,y) \to P(x,y)$	(t10)	$SC(x) \leftrightarrow \forall y, z(x = y + z \to C(y,z))$	(t11)
$\exists z(SC(z) \wedge O(z,x) \wedge O(z,y) \wedge \forall w(P(w,z) \to (O(w,x) \vee O(w,y)))) \to C(x,y)$			(t12)
$z = \sum x \phi x \to \forall y(C(y,z) \to \exists x(\phi x \wedge C(y,x)))$			(t13)
$P(x, cx)$	(t14)	$c(cx) = cx$	(t15)
$c(x+y) = cx + cy$	(t16)	$cx =_{df} \sim (ex)$	(t17)
$ex =_{df} i(\sim x)$	(t18)	$ix =_{df} \sum z \forall y(C(z,y) \to O(x,y))$	(t19)
Additional axioms, definitions, and theorems			
$PP(x,y) =_{df} P(x,y) \wedge \neg P(y,x)$	(t20)	$O(x,y) =_{df} \exists z(P(z,x) \wedge P(z,y))$	(t21)
$EQ(x,y) =_{df} P(x,y) \wedge P(y,x)$	(t22)	$TPP(x,y) =_{df} PP(x,y) \wedge \neg IPP(x,y)$	(t23)
$IPP(x,y) =_{df} PP(x,y) \wedge \forall z(C(z,x) \to O(z,y))$			(t24)
$\neg PP(x,x)$	(t25)	$PP(x,y) \wedge PP(y,z) \to PP(x,z)$	(t26)
$PP(x,y) \to \neg PP(y,x)$	(t27)		

Kuratowski General Extensional MereoTopology

Core axioms and definitions			
$P(x,x)$	(t1)	$P(x,y) \wedge P(y,z) \rightarrow P(x,z)$	(t2)
$P(x,y) \wedge P(y,x) \rightarrow x = y$	(t3)	$\neg P(y,x) \rightarrow \exists z(P(z,y) \wedge \neg O(z,x))$	(t4)
$\exists w \phi(w) \rightarrow \exists z \forall w(O(w,z) \leftrightarrow \exists v(\phi(v) \wedge O(w,v)))$			(t5)
$C(x,x)$	(t6)	$C(x,y) \rightarrow C(y,x)$	(t7)
$P(x,y) \rightarrow E(x,y)$	(t8)	$E(x,y) =_{df} \forall z(C(z,x) \rightarrow C(z,y))$	(t9)
$E(x,y) \rightarrow P(x,y)$	(t10)	$SC(x) \leftrightarrow \forall y,z(x = y + z \rightarrow C(y,z))$	(t11)
$\exists z(SC(z) \wedge O(z,x) \wedge O(z,y) \wedge \forall w(P(w,z) \rightarrow (O(w,x) \vee O(w,y)))) \rightarrow C(x,y)$			(t12)
$z = \sum x \phi x \rightarrow \forall y(C(y,z) \rightarrow \exists x(\phi x \wedge C(y,x)))$			(t13)
$P(x,cx)$	(t14)	$c(cx) = cx$	(t15)
$c(x+y) = cx + cy$	(t16)	$cx =_{df} \sim(ex)$	(t17)
$ex =_{df} i(\sim x)$	(t18)	$ix =_{df} \sum z \forall y(C(z,y) \rightarrow O(x,y))$	(t19)
Additional axioms, definitions, and theorems			
$PP(x,y) =_{df} P(x,y) \wedge \neg P(y,x)$	(t20)	$O(x,y) =_{df} \exists z(P(z,x) \wedge P(z,y))$	(t21)
$EQ(x,y) =_{df} P(x,y) \wedge P(y,x)$	(t22)	$TPP(x,y) =_{df} PP(x,y) \wedge \neg IPP(x,y)$	(t23)
$IPP(x,y) =_{df} PP(x,y) \wedge \forall z(C(z,x) \rightarrow O(z,y))$			(t24)
$\neg PP(x,x)$	(t25)	$PP(x,y) \wedge PP(y,z) \rightarrow PP(x,z)$	(t26)
$PP(x,y) \rightarrow \neg PP(y,x)$	(t27)		