A comparison of different types of declarative modelling

Maria Keet

Department of Computer Science, University of Cape Town, South Africa
mkeet@cs.uct.ac.za

HydroSciences Montpellier
Université de Montpellier & Institut de Recherche pour le Développement
20 June 2024

Outline

Outline

Models galore

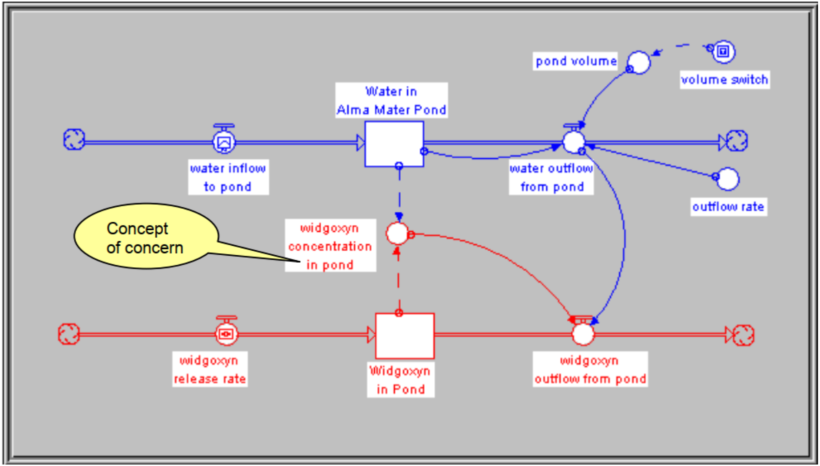
- Physical model; e.g., Lego brick house
- Mathematical model; e.g., climate change model, bacterial growth in cheese-making
- Machine learning & cs.; e.g., data-driven spellchecker, LLM
- Conceptual models; e.g., concept maps, UML diagrams, ontologies

Models galore

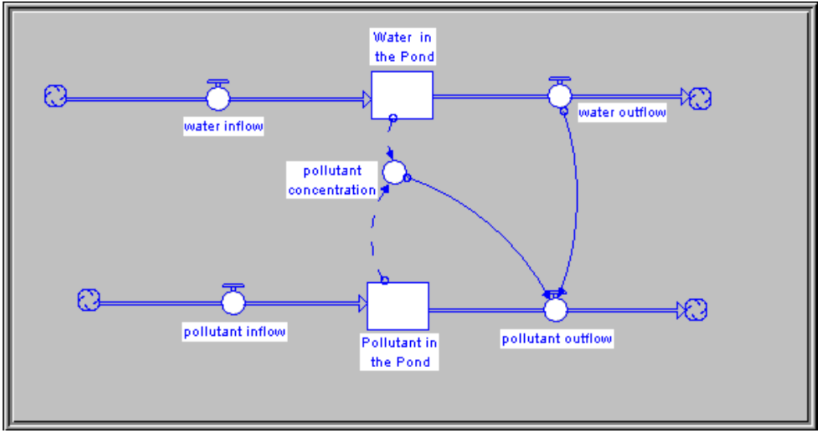
- Physical model; e.g., Lego brick house
- Mathematical model; e.g., climate change model, bacterial growth in cheese-making
- Machine learning & cs.; e.g., data-driven spellchecker, LLM
- Conceptual models; e.g., concept maps, UML diagrams, ontologies

⇒ Gentle, mostly non-technical introduction to conceptual models in computing, with a few half-baked props on 'things with water'

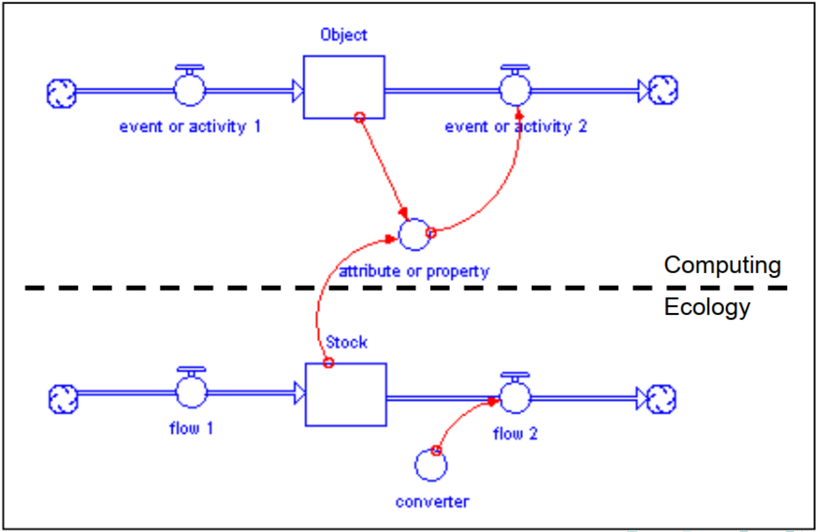
An old TOY example – linking different perspectives



An old TOY example – linking different perspectives



An old TOY example – linking different perspectives



Matching things and terminology (1/2)

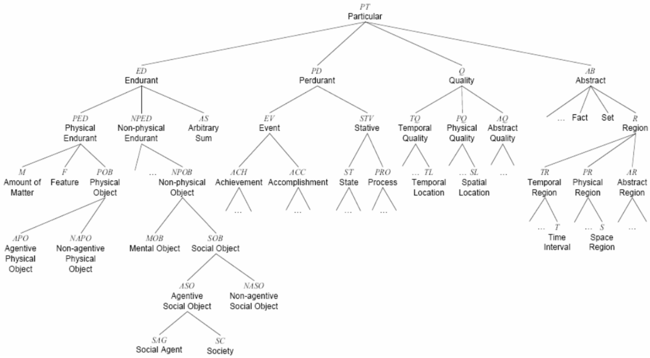
- *Stock* typically named with a noun (particular or universal)
- *Flow* named with a verb
- *Converter* as an attribute related to Flow or Stock
- *Action Connector* relates the former

- *Object* is candidate for an endurant
- *Event_or_activity* for a method or perdurant
- *Converter* maps to attribute_or_property
- *Action Connector* candidate for relationship between any two of Flow, Stock and Converter

More information and evaluation (with the microbial loop in ocean): Keet CM. Factors affecting ontology development in ecology. DILS'05.

Matching things and terminology (1/2)

- $\forall x(Stock(x) \rightarrow EnDurant(x))$
- $\forall x(Flow(x) \rightarrow PerDurant(x))$
- $\forall x(Converter(x) \rightarrow (Quality(x) \vee State(x)))$
- $\forall x(ActionConnector(x) \rightarrow Relationship(x))$



DOLCE: [Masolo et al.(2003)]

Some questions (part 1)

- Stella is rather outdated; which other (declarative) modelling languages could this be done with?
- What to convert it into?
    - Domain specific language
    - Conceptual data model, process model, ...
    - More generic terminology, classes, relations, and constraints (an ontology?)

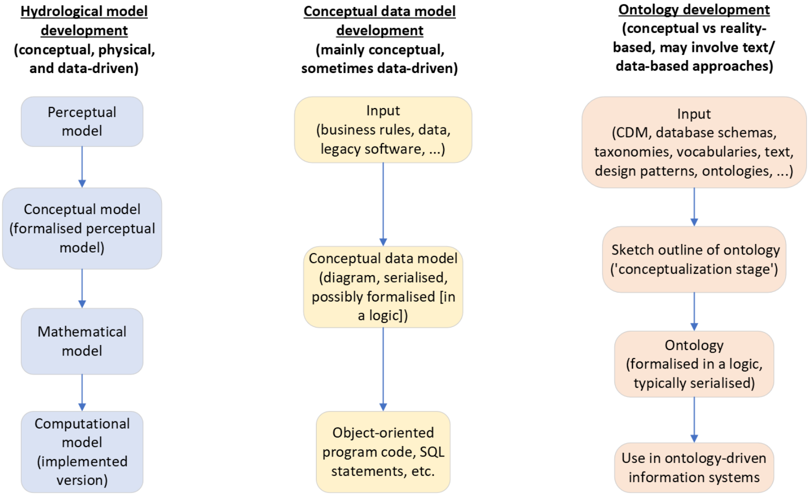
Some questions (part 1)

- Stella is rather outdated; which other (declarative) modelling languages could this be done with?
- What to convert it into?
    - Domain specific language
    - Conceptual data model, process model, ...
    - More generic terminology, classes, relations, and constraints (an ontology?)
- Why even do this?

Some questions (part 1)

- Stella is rather outdated; which other (declarative) modelling languages could this be done with?
- What to convert it into?
  - Domain specific language
  - Conceptual data model, process model, ...
  - More generic terminology, classes, relations, and constraints (an ontology?)
- Why even do this?
  - integrating models, comparing models, checking for possible inconsistencies automatically
  - Management of the mathematical models: separate declarative from the imperative

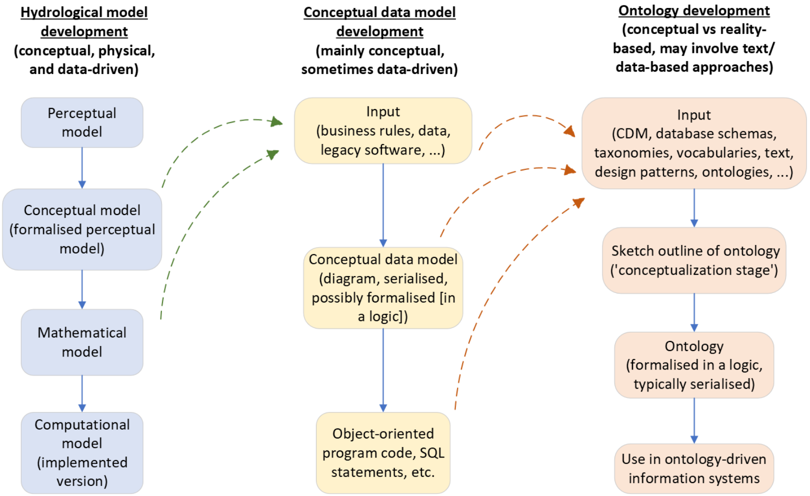
Terminology, simplified processes, and some links



---

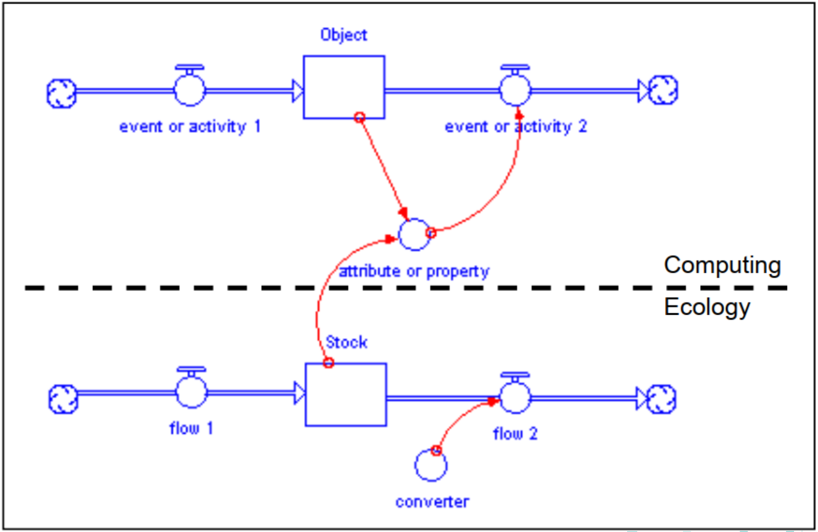
hydrological modelling information from [Solomatine and Wagener(2011)]

Terminology, simplified processes, and some links



hydrological modelling information from [Solomatine and Wagener(2011)]

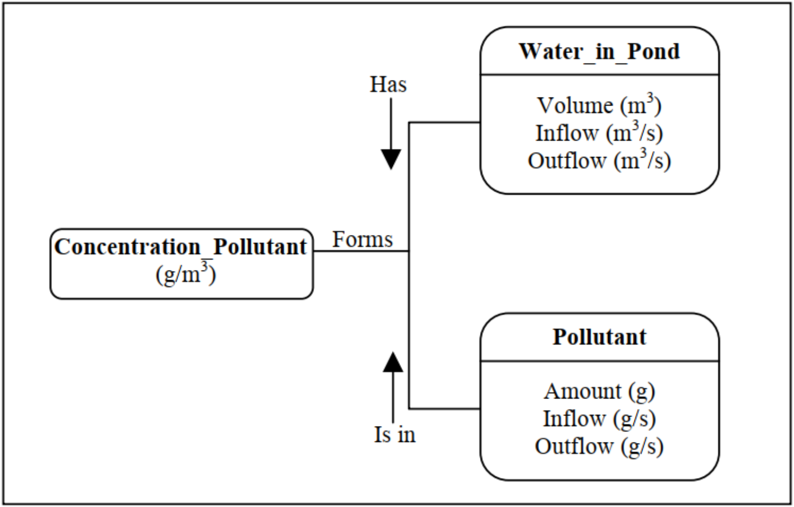
On the maths – scaffold generation



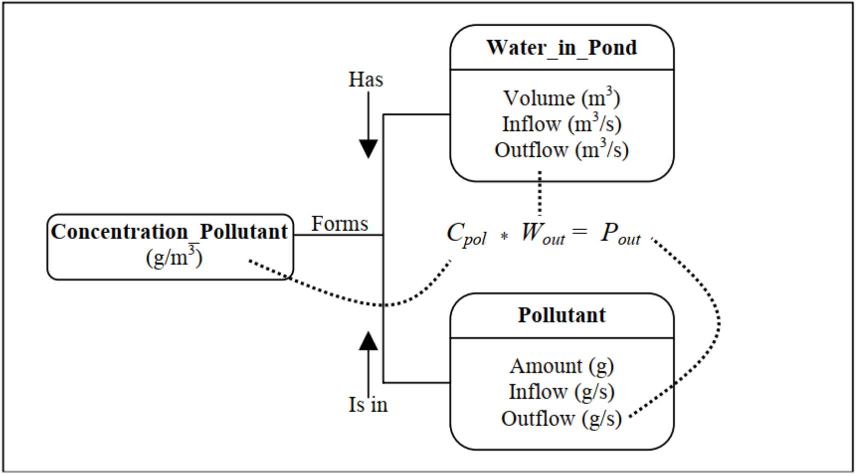
On the maths – scaffold generation

```
Object(t) = Object(t - dt) + (event_or_activity_1 - event_or_activity_2) * dt
INIT Object = { Place initial value here... }
 INFLOWS:
    event_or_activity_1 = { Place right hand side of equation here... }
 OUTFLOWS:
    event_or_activity_2 = { Place right hand side of equation here... }
Stock(t) = Stock(t - dt) + (flow_1 - flow_2) * dt
INIT Stock = { Place initial value here... }
 INFLOWS:
    flow_1 = { Place right hand side of equation here... }
 OUTFLOWS:
    flow_2 = { Place right hand side of equation here... }
attribute_or_property = { Place right hand side of equation here... }
converter = { Place right hand side of equation here... }
```

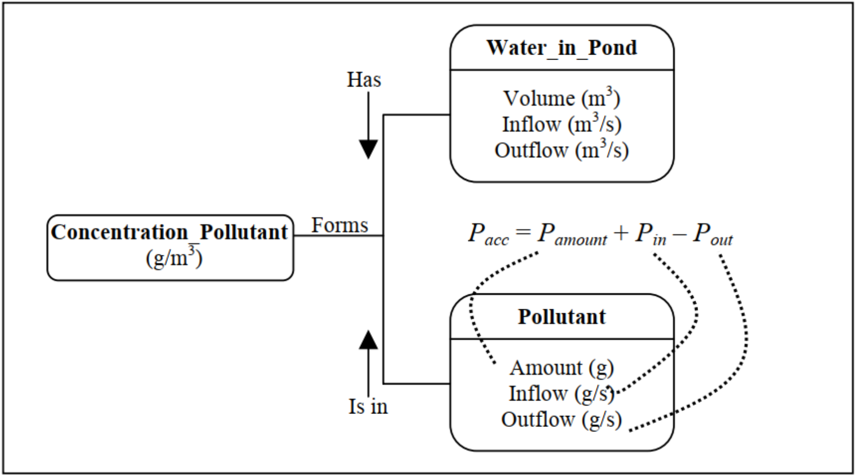
On the maths – scaffold generation



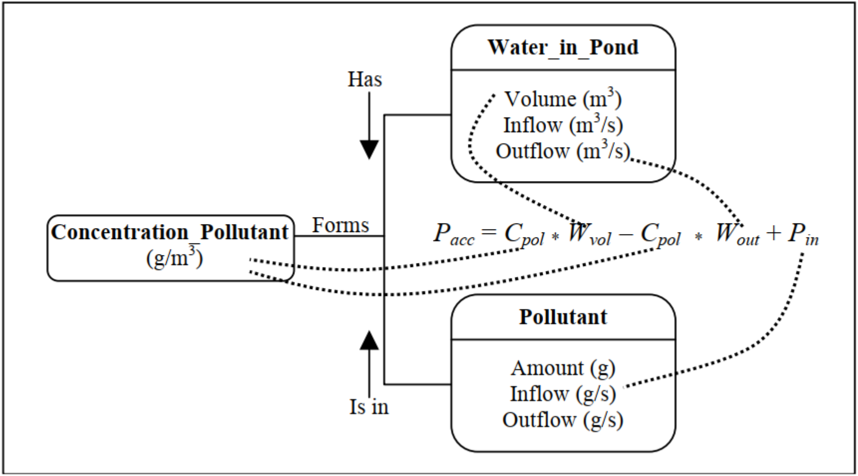
On the maths – scaffold generation



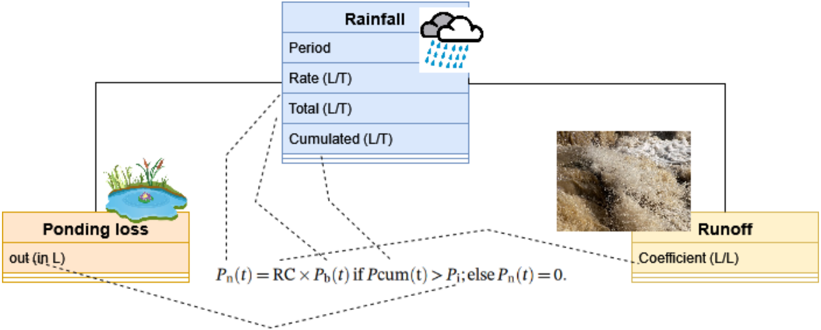
On the maths – scaffold generation



On the maths – scaffold generation



On the maths – a recent paper



formula from: Chahinian N, et al. (2023). Evaluation of an early flood warning system in Bamako (Mali): Lessons learned from the flood of May 2019. J Flood Risk Mgmt.

Some questions (part 2)

- How to do something like this systematically?
- Will any of the current applied maths markup languages (e.g., SBML) suffice?

Some questions (part 2)

- How to do something like this systematically?
- Will any of the current applied maths markup languages (e.g., SBML) suffice?
- Why look into this?

Some questions (part 2)

- How to do something like this systematically?
- Will any of the current applied maths markup languages (e.g., SBML) suffice?
- Why look into this?
  - Annotation (declarative representation) of the formula separates the what (the formulae) from the how (the code how to compute it)
  - One declarative formula/model can be loaded ‘trivially’ across programming languages and executed
  - (Parts of) The declarative formula/model can more easily be reused, saving design time, facilitating interoperability

SBML: random example of an SBML model from the BioModels database

BIOMD0000000531_url.xml

```
      <listOfUnitDefinitions>
      <unitDefinition id="volume" metaid="_224f94f3-7d8c-4058-899c-ef158671d01e" name="volume">
        <listOfUnits>
          <unit exponent="1" kind="litre" metaid="b0b05891-9c17-4104-836c-e37a86b35f95" multiplier="1" scale="-3"/>
        </listOfUnits>
      </unitDefinition>
      <unitDefinition id="time" metaid="_45409b69-6247-4659-b564-97ab08e0e5cd" name="time">
        <listOfUnits>
          <unit exponent="1" kind="second" metaid="e38bd50b-5dda-4daa-a633-c6f6c51d2ed8" multiplier="3600" scale="0"/>
        </listOfUnits>
      </unitDefinition>
      <unitDefinition id="substance" metaid="_7abe9aca-312c-4b5d-baa2-c64840864da1" name="substance">
        <listOfUnits>
          <unit exponent="1" kind="mole" metaid="_5aa1de13-c181-4205-a36f-9fbe53894ab3" multiplier="1" scale="-3"/>
        </listOfUnits>
      </unitDefinition>
```

SBML: random example of an SBML model from the BioModels database

BIOMD0000000531_url.xml

```
    <listOfParameters>
      <parameter constant="true" id="Ka" metaid="COPASI2" name="Ka" value="1.44">
        <annotation>
          <COPASI xmlns="http://www.copasi.org/static/sbml">
            <rdf:RDF xmlns:dcterms="http://purl.org/dc/terms/" xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
              <rdf:Description rdf:about="#COPASI2">
                <dcterms:created>
                  <rdf:Description>
                    <dcterms:W3CDTF>2014-07-17T15:42:40Z</dcterms:W3CDTF>
                  </rdf:Description>
                </dcterms:created>
              </rdf:Description>
            </rdf:RDF>
          </COPASI>
        </annotation>
            </parameter>
      <parameter constant="true" id="kb" metaid="COPASI3" name="kb" value="1.6E-10">
        <annotation>
```

SBML: random example of an SBML model from the BioModels database

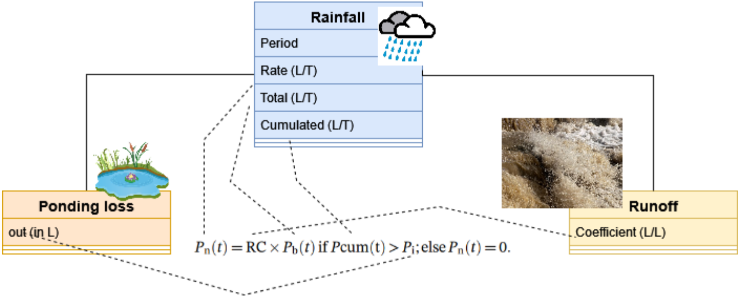
BIOMD0000000531_url.xml

```
      <listOfRules>
        <assignmentRule metaid="_15d0c074-0652-4ed7-b795-9a1fd1a61655" variable="kat50">
          <math xmlns="http://www.w3.org/1998/Math/MathML">
            <apply>
              <ln/>
              <apply>
                <plus/>
                <apply>
                  <divide/>
                  <cn> 1 </cn>
                  <ci> kb </ci>
                </apply>
                <cn> 1 </cn>
              </apply>
            </apply>
          </math>
              </assignmentRule>
        <assignmentRule metaid="d9080aa4-e7cc-4a7a-b7f7-b7ee59435fb4" variable="v50t50">
```

Limitations of the domain models

- There are very many notations to learn
- Objects and arrows, but no way to specify how many outgoing arcs there may be
- Limited computational use among models
- Proliferation of incompatible modelling tools that are cumbersome to maintain

Rainfall questions – loose ends in the model



- What are the names of the relations? the constraints? The relevant attributes? And their data types?

Outline

Solutions to limitations of the domain models

- Devise one notation for all
- More expressive (more features) than only objects and arrows
- Computational support
- (Try to) Standardise to make tooling development 'economically' viable

Solutions to limitations of the domain models

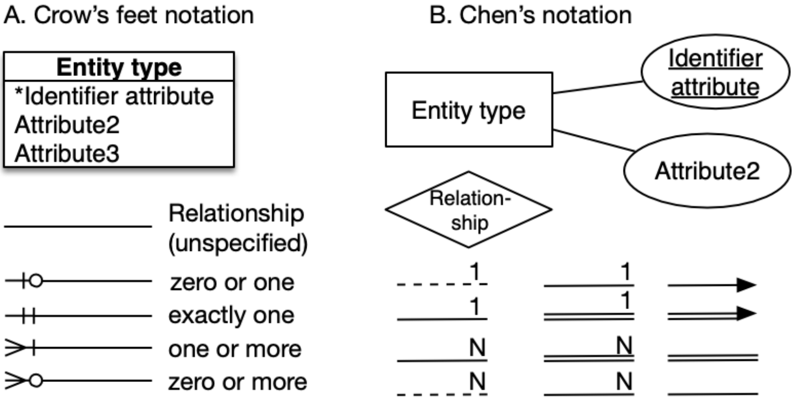
- Devise one notation for all
- More expressive (more features) than only objects and arrows
- Computational support
- (Try to) Standardise to make tooling development 'economically' viable

$\Rightarrow$ Conceptual data models

Conceptual data models

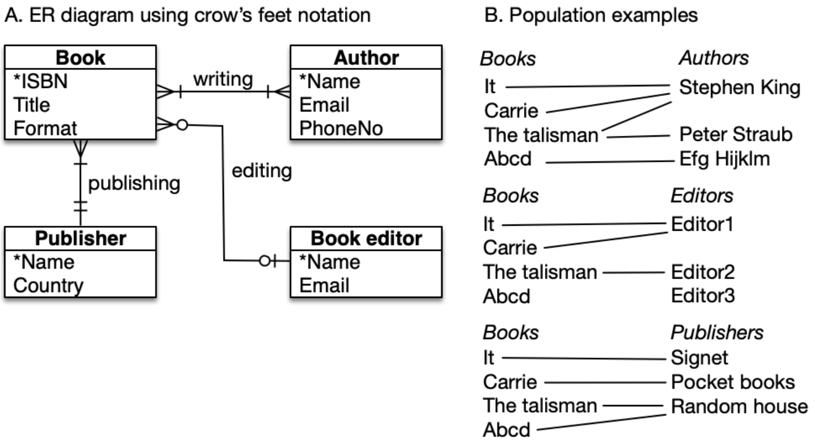
- A class of models that capture the information about the data that are to be stored in the prospective software system (and possibly manipulated)
- There are several conceptual data modelling language families and notations

Conceptual data models – Example of language elements

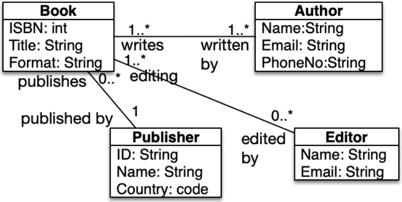


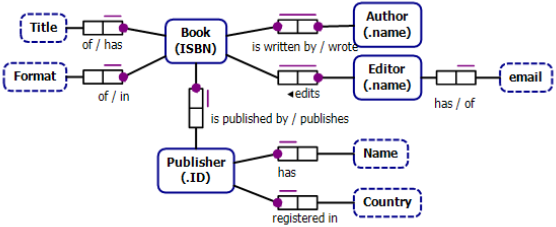
Note: no 'stock', 'flow' etc, but type of element for any subject domain

Conceptual data models – Example



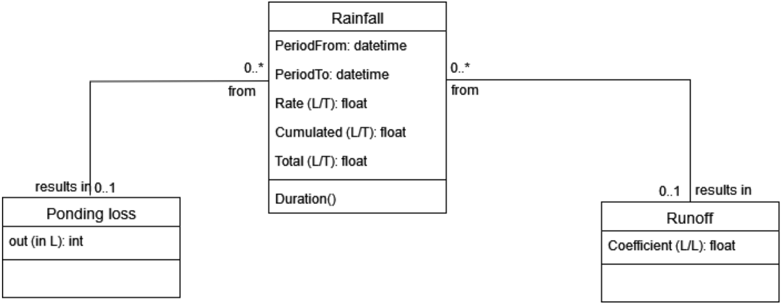
Same example in UML and ORM





Convert automatically between EER, UML, ORM, choose preferred notation: Braun G, Fillottrani PR, Keet CM. A Framework for Interoperability Between Models with Hybrid Tools, J of Intelli. Inf. Sys. 2023.

Rainfall, again — UML style



- Convert UML class diagram (semi-)automatically into program code
- Easier to communicate with other domain experts and programmers what's in the code
- Easier to reuse with other math formula that use same entities

Limitations of conceptual data model models in theory or practice

- For one specific application only – need to re-do it for each application, integration issues
- Solutions to recurring modelling issues re-invented time and again (and same mistakes made)
- Mostly informal diagrams that suffer from ambiguity (intentionally or not)
- Limited authoring guidelines[1]
- Some quality control mechanisms

[1] Mainly the CSDP for ORM [Halpin(2001)]; an example for EER: `http://www.meteck.org/modellingbook/DanceSchoolExample.html`; TDD proposal for UML [Tort and Olivé(2010)]

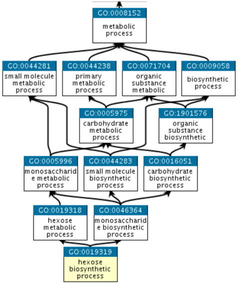
Outline

Solving limitations of conceptual data model models with ontologies

- Model for a subject domain, of use across multiple applications for use, reuse, integration
- Provides solutions to recurring modelling issues, saves re-inventing
- Logic-based, as precise as permitted within the language
- Multiple quality control mechanisms (theories, methods, techniques, tools)

An ontology...

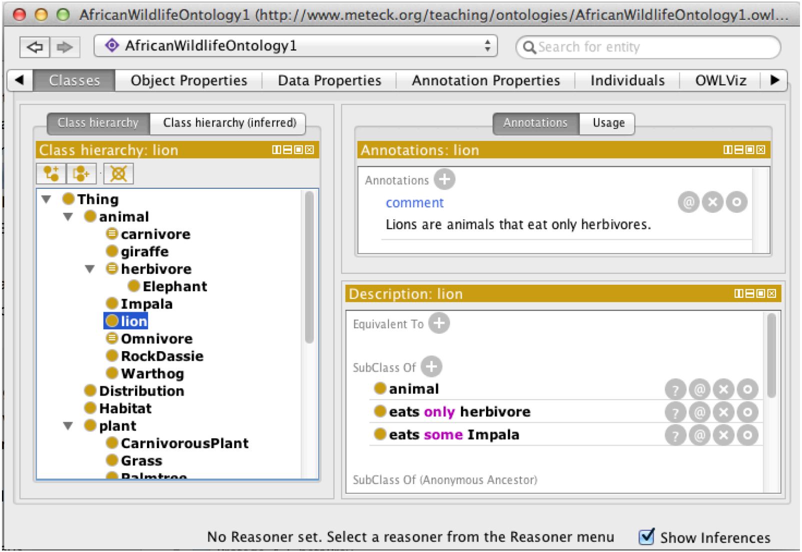
Simplified graphical rendering of a fragment of most popular one:



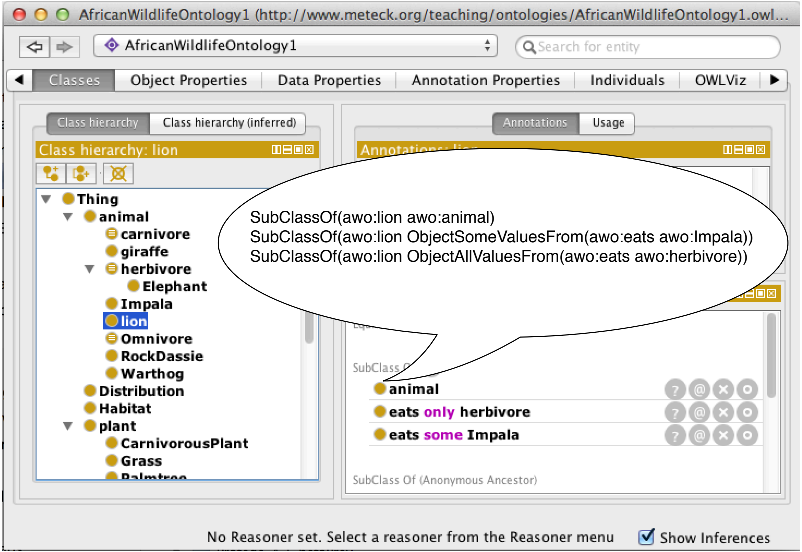
http://geneontology.org/docs/ontology-documentation/

(informally) *an* ontology is an engineering artefact in machine-processable format, which contains the entity types, their relationships, and constraints that hold over them of a particular (subject) domain.

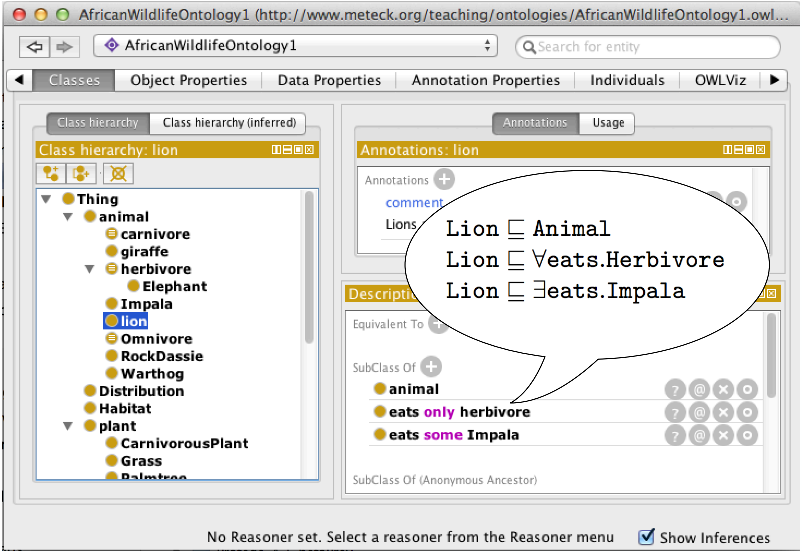
In an ontology development environment (ODE)...



... happenings behind the GUI ...



… and underlying that serialisation



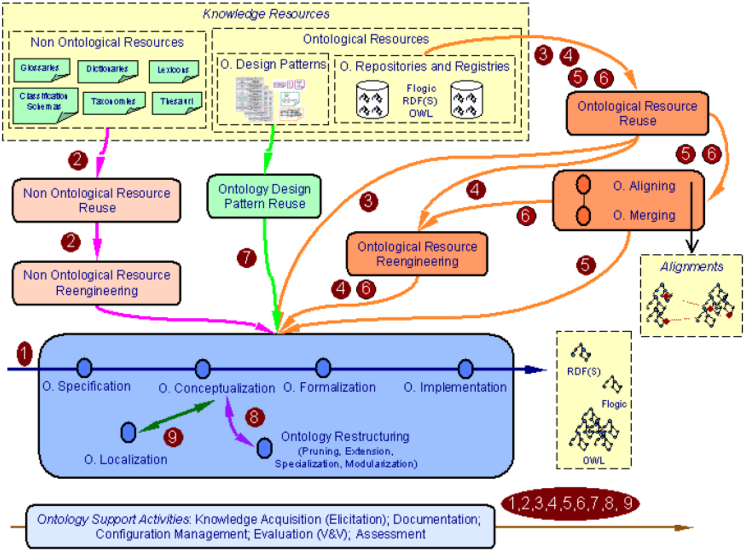
Why ontologies?

- For their own sake, possible future use
- Representing a scientific theory precisely
- Facilitating communication among humans
- Enabling communication between software applications or modules in a complex system
- Used for and in many different ontology-driven information systems: a.o., data integration, recommender systems, NLP, textbook enhancements, Q&A systems)

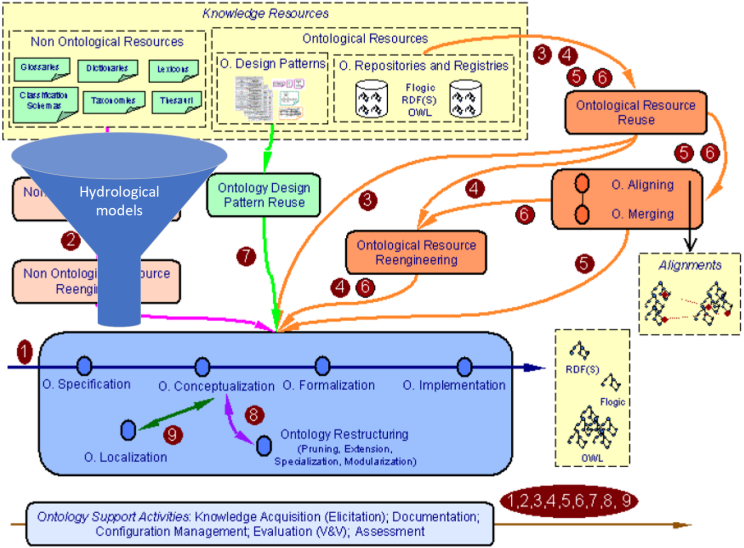
Why ontologies?

- For their own sake, possible future use
- Representing a scientific theory precisely
- Facilitating communication among humans
- Enabling communication between software applications or modules in a complex system
- Used for and in many different ontology-driven information systems: a.o., data integration, recommender systems, NLP, textbook enhancements, Q&A systems)
- Examples: finding 'new' knowledge (bio-chemistry) [Wolstencroft et al.(2007)] save research time (ecology) [Madin et al.(2008)] semantic comparisons of text (healthcare) [Reese et al.(2023)], energy-optimised building system control [Pruvost and Enge-Rosenblatt(2022)] etc.

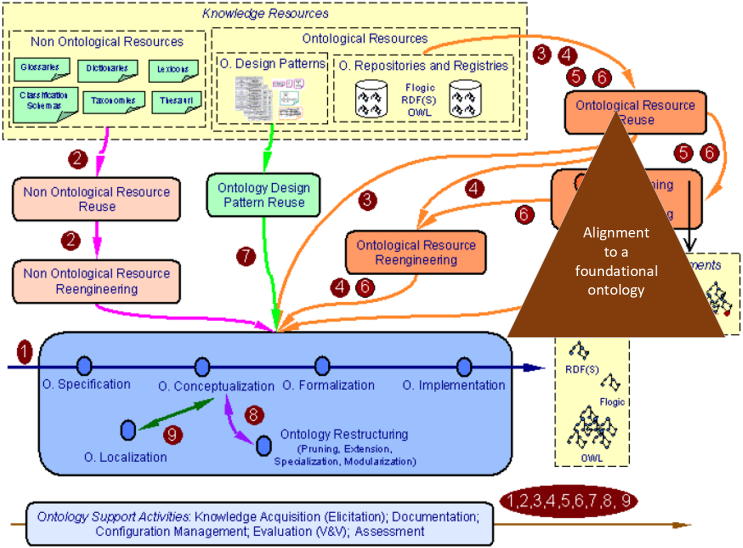
Scenarios for building Ontology Networks (NEON methodology)



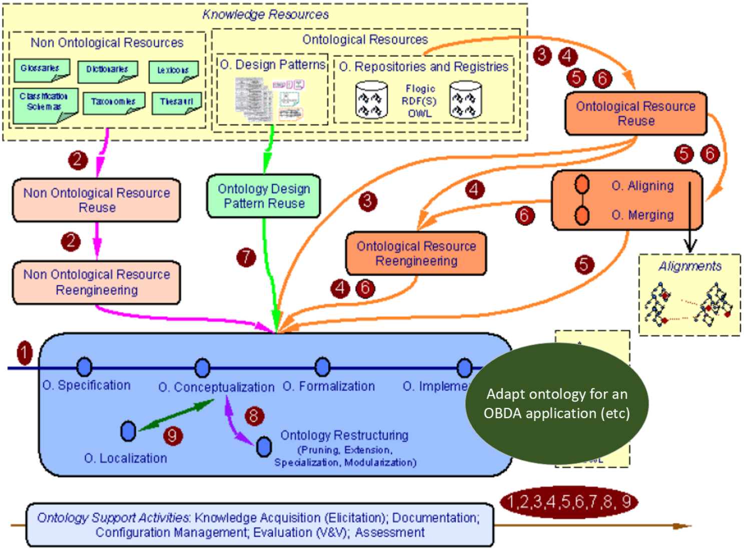
Scenarios for building Ontology Networks (NEON methodology)



NeOn – some extensions



NeOn – some extensions



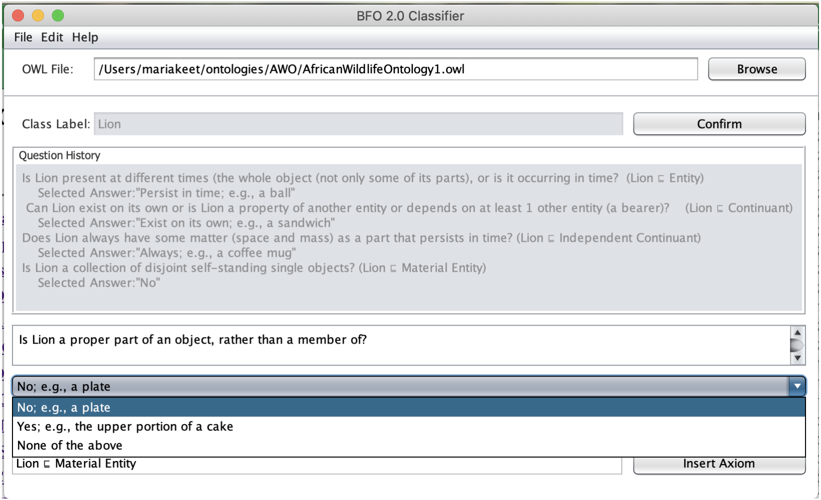
Ontology development at the ‘micro-level’ level (cf. macro)

- Figure out what to add: which vocabulary and axioms
- Need to get those axioms into the ontology somehow

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

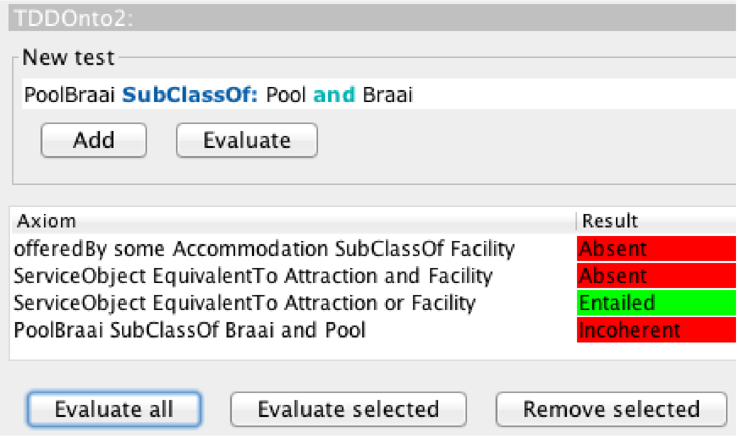
- Figure out what to add: which vocabulary and axioms
- Need to get those axioms into the ontology somehow
- The actual modelling, or *ontology authoring*, using micro-level guidelines, methods, and tools
  - Methods, such as reverse engineering and text mining to start, OntoClean and OntoParts to improve an ontology's quality
  - Tools to model, to reason, to debug, to integrate, to link to data

Align to it – manually or assisted with D3 or BFOClassifier



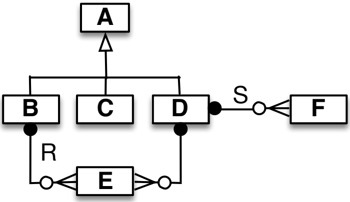
Bernabé C, Keet CM, Khan ZC, Mahlaza Z. A method to improve alignments between domain and foundational ontologies. FOIS'23.

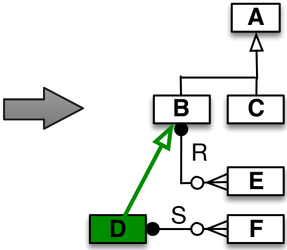
Test-driven development



Davies K, Keet CM, Lawrynowicz A. More Effective Ontology Authoring with Test-Driven Development and the TDDonto2 tool. Int J on Artif Intell Tools, 2019.

A note on that automated reasoning – Illustration



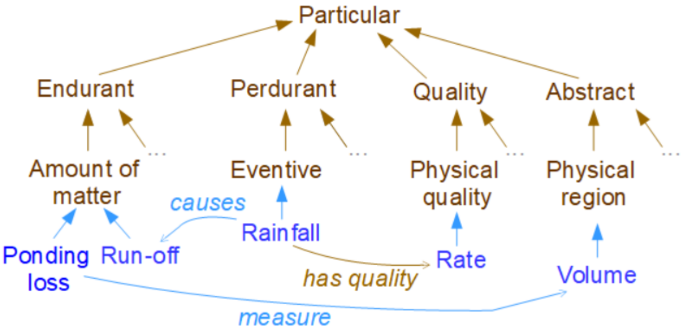


$B \sqsubseteq A$

$C \sqsubseteq A \qquad B \sqsubseteq \exists R.E$

$D \sqsubseteq A \qquad D \sqsubseteq \exists R.E \sqcap \exists S.F$

Rainfall: sample *sketch* and some axioms for an ontology



$Rainfall \sqsubseteq Eventive$ ('rainfall is an event'), $Rainfall \sqsubseteq \exists causes.Runoff$ ('each rainfall event causes some amount of runoff'), $PondingLoss \sqsubseteq = 1\, measure.Volume$ ('each ponding loss has exactly one measure of volume' (of the loss)), ...

Limitations of ontologies?

Limitations of ontologies?

- Yes, there are...
- ... but not part of this talk

Feature-based comparison

Table 7.1 Comparison of types of models along a set of properties

Model type	Feature: Main aim or function	Where used (mainly)	Development methodologies	Software assistance	Language freedom	Precision
Mind Maps	Basic structuring of a topic	Education, business	A little	Yes, many drawing tools	Limited	Low
Biology models	Visualise biology knowledge (structures and processes)	Biology research, textbooks	No	Drawing tools, some runtime usage (simulations)	Ranges from self-imposed to complete freedom	Low/ medium
Conceptual data models	Capture characteristics of data to be stored and processed in an program	Analysis and design stage of database and program development	Yes	Drawing tools, limited runtime usage	Ranges from standardised languages to partial freedom to design one	Medium
Ontologies	Represent knowledge of a subject domain precisely and in a computer processable way	Computing and IT (Data integration, Enterprise systems, Web search, etc.)	Yes, many	Editors (diagram, text), runtime usage	Ranges from standardised languages to partial freedom to design one	High (but medium/ low is possible)
Ontology	Characterise one small aspect of interest precisely and in much detail	Research	No	No	Yes, can define as one goes along	High/ Very high

(also from the 'what and how' book)

Outline

Recap and future work

- Representing knowledge of the domain, semantics of the equations, declarative-imperative separation
- Different types of declarative models: diagrams, conceptual data models, ontologies
- Which one suits best depends on the task

Recap and future work

- Representing knowledge of the domain, semantics of the equations, declarative-imperative separation
- Different types of declarative models: diagrams, conceptual data models, ontologies
- Which one suits best depends on the task
- Opportunities for declarative hydrological modelling
- Repurpose some of the design steps from computing to hydrological modelling?
- Use the resulting models for ontology and conceptual model development

Acknowledgments

- Collaborators on some of this work: Pablo Fillottrani (Universidad Nacional del Sur, Argentina), Zubeida Khan (CSIR, South Africa), Agnieszka Ławrynowicz (PUT, Poland), César Bernabé (LUMC, The Netherlands), Zola Mahlaza (UCT)
- Current and former students on some of this work: Kieren Davies
- Funding of this secondment to HSM:

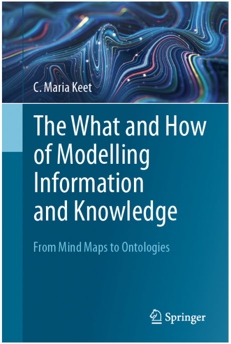
STARWARS

Funded by
the European Union

Thank you!

Questions?

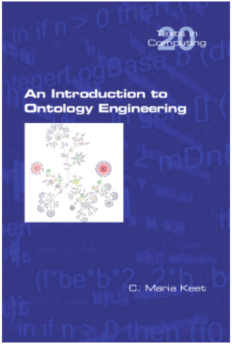
- My book on modelling,
- aimed at a broader audience, and
- available in hardcopy and eBook
- https://link.springer.com/book/10.1007/978-3-031-39695-3



Thank you!

Questions?

- My textbook on ontology engineering (aimed at computer scientists)
- Free pdf (and slides and exercises) at `https://people.cs.uct.ac.za/~mkeet/OEbook/`
- Also available in paperback (College Publications):



References I

T.A. Halpin.
Information Modeling and Relational Databases.
San Francisco: Morgan Kaufmann Publishers, 2001.

Joshua S. Madin, Shawn Bowers, Mark P. Schildhauer, and Matthew B. Jones.
Advancing ecological research with ontologies.
Trends in Ecology & Evolution, 23(3):159–168, 2008.

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.

Hervé Pruvost and Olaf Enge-Rosenblatt.
Using Ontologies for Knowledge-Based Monitoring of Building Energy Systems, pages 762–770.
doi: 10.1061/9780784483893.094.

J. Reese et al.
Generalisable long covid subtypes: findings from the nih n3c and recover programmes.
eBioMedicine, 87(104413), 2023.

DP Solomatine and T. Wagener.
Treatise on Water Science, Vol 2, chapter Hydrological Modeling, pages 435–457.
Elsevier, 2011.
doi: 10.1016/B978-0-444-53199-5.00044-0.

References II

Mari Carmen Suarez-Figueroa, Guadalupe Aguado de Cea, Carlos Buil, Klaas Dellschaft, Mariano Fernandez-Lopez, Andres Garcia, Asuncion Gómez-Pérez, German Herrero, Elena Montiel-Ponsoda, Marta Sabou, Boris Villazon-Terrazas, and Zheng Yufei.
NeOn methodology for building contextualized ontology networks.
NeOn Deliverable D5.4.1, NeOn Project, 2008.

Albert Tort and Antoni Olivé.
An approach to testing conceptual schemas.
Data & Knowledge Engineering, 69:598–618, 2010.

K. Wolstencroft, R. Stevens, and V. Haarslev.
Applying OWL reasoning to genomic data.
In C.J.O. Baker and H. Cheung, editors, *Semantic Web: revolutionizing knowledge discovery in the life sciences*, pages 225–248. Springer: New York, 2007.