Ontologies for Security and Privacy

Lecture 4 – Computational Ontologies (Part 2)

Alessandro Oltramari

CyLab, Carnegie Mellon University

Carnegie Mellon

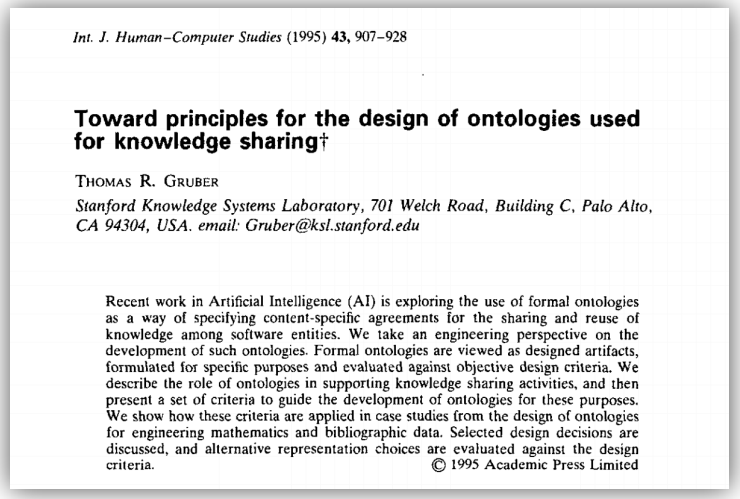
Definitions and languges

RECAP

Carnegie Mellon

"An ontology is a formal specification of a conceptualization"

- What is a formal specification?
  - An expression in a rigorous language
    - Formal Syntax & Semantics
- What is a conceptualization?
  - By and large, it's a mental model of a situation/domain/world



Int. J. Human–Computer Studies (1995) **43**, 907–928

Toward principles for the design of ontologies used for knowledge sharing†

THOMAS R. GRUBER

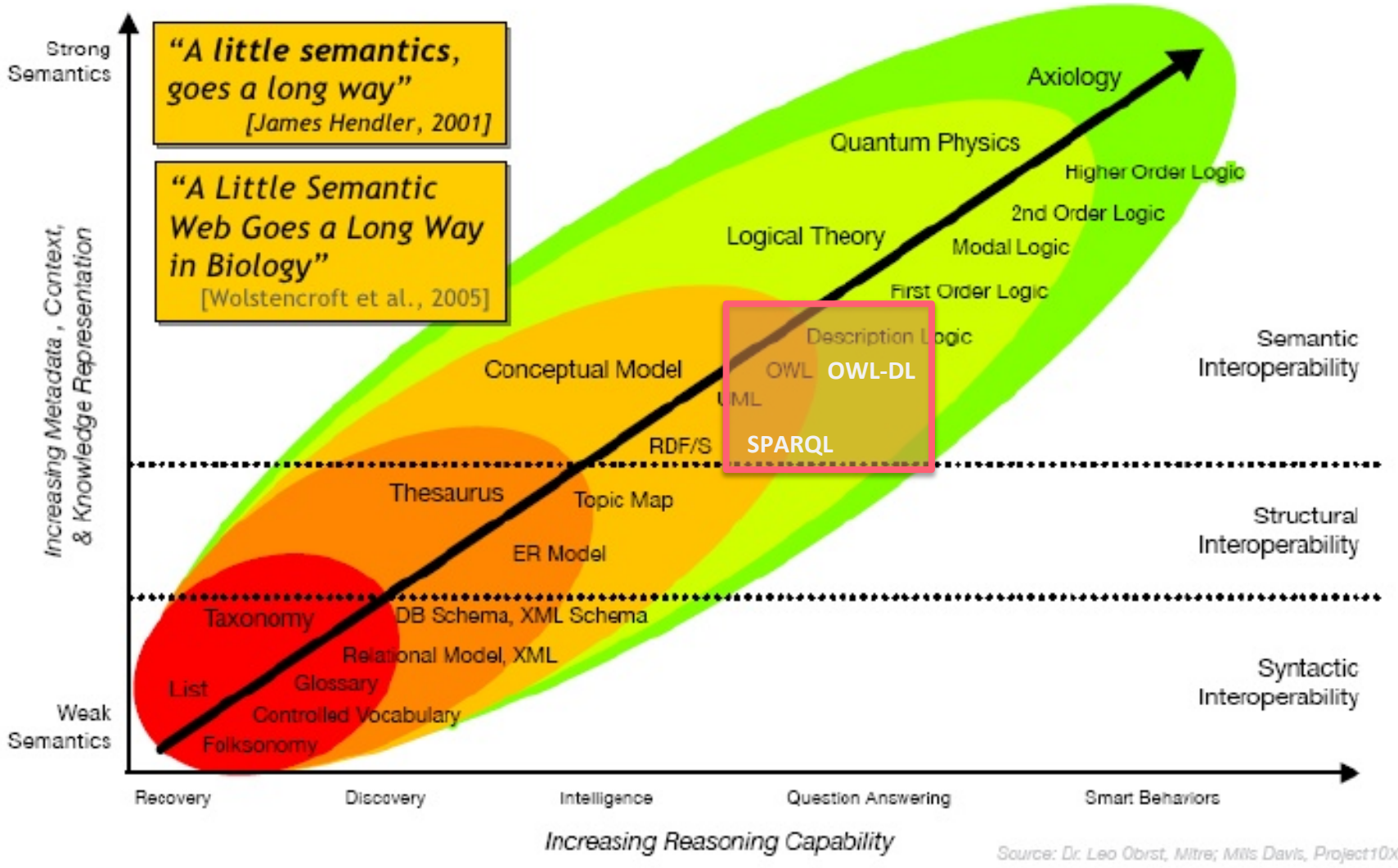
Stanford Knowledge Systems Laboratory, 701 Welch Road, Building C, Palo Alto, CA 94304, USA. email: Gruber@ksl.stanford.edu

Recent work in Artificial Intelligence (AI) is exploring the use of formal ontologies as a way of specifying content-specific agreements for the sharing and reuse of knowledge among software entities. We take an engineering perspective on the development of such ontologies. Formal ontologies are viewed as designed artifacts, formulated for specific purposes and evaluated against objective design criteria. We describe the role of ontologies in supporting knowledge sharing activities, and then present a set of criteria to guide the development of ontologies for these purposes. We show how these criteria are applied in case studies from the design of ontologies for engineering mathematics and bibliographic data. Selected design decisions are discussed, and alternative representation choices are evaluated against the design criteria.

Therefore, *building an **ontology** is essentially the task of **expressing** a mental model of a situation/domain/world using a formal language*

Carnegie Mellon

Expressivity, Reasoning, Interoperability



Class

- In the following we introduce some examples of OWL core components; most are taken from the ontology pizza.owl (we'll see how this ontology looks like in the Protégé-OWL platform).

```
<owl:Class rdf:resource="AnchoviesTopping">
  <rdfs:subClassOf rdf:resource="#FishTopping"/>
  <owl:disjointWith rdf:resource="#PrawnsTopping"/>
  <owl:disjointWith rdf:resource="#MixedSeafoodTopping"/>
</owl:Class>
```

- OWL extends RDF: it adopts RDF meaning of classes and properties, extending the expressivity with its own primitives

  `owl:Thing` → the most general class (it subsumes every class);

  `owl:Nothing` → the empty class (every class subsumes it)

Reminder – layers of languages

XML→XML(S)→RDF→RDF(S)→OWL

Carnegie Mellon

Object Property

- Object property relates classes to other classes:

```
<owl:ObjectProperty rdf:about="#hasTopping">
  <rdfs:subPropertyOf>
    <owl:ObjectProperty rdf:about="#hasIngredient"/>
  </rdfs:subPropertyOf>
  <rdfs:domain rdf:resource="#Pizza"/>
  <rdfs:range rdf:resource="#PizzaTopping"/>
  <owl:inverseOf>
    <owl:ObjectProperty rdf:about="isToppingOf"/>
  </owl:inverseOf>
</owl:ObjectProperty>
```

InverseOf interchange domain with range

Carnegie Mellon

Datatype Property

- Datatype property relates classes (objects) to datatype values:

```
<owl:DataProperty rdf:about="#Table_number">
   <rdfs:range rdf:resource=
   "http://www.w3.org/2001/XMLschema#nonNegativeInteger"/
   >
  </owl:DataProperty>
```

OWL does not provide predefined datatypes: it uses XML Schema datatypes

Basic features

OWL 2

Carnegie Mellon

OWL 2: Negative Properties

- While OWL 1 provides means to assert values of a property for an individual, it does not provide a construct for directly asserting values that an individual does not have (negative facts). OWL 2 does it.

- `NegativeObjectPropertyAssertion` & `NegativeDataPropertyAssertion` state that a given property does not hold for the given individuals.

  - NegativeObjectPropertyAssertion (*:IsLocated :ThisFile :MyServer*)

Carnegie Mellon

Extra Datatypes and Datatype Restrictions

OWL 1 provides support for only integers and strings as datatypes and does not support any subsets of these datatypes.

E.g.: we could state that every person has an age, which is an integer, but could not restrict the range of that datatype to say that adults have an age greater than 18. OWL 2 provides this capability

```
SubClassOf(:Teenager
      DataSomeValuesFrom (:hasAge
        DatatypeRestriction(xsd:integer
                            xsd:minExclusive "12"^^xsd:integer
                            xsd:maxInclusive "19"^^xsd:integer
                            )
                          )
           )
```

Carnegie Mellon

Keys

- An HasKey axiom states that each named instance of a class is uniquely identified by a (data or object) property or a set of properties - that is, if two named instances of the class coincide on values for each of key properties, then these two individuals are the same

- Consider the ontology consisting of the following axioms.

HasKey(*owl:Person* () (*a:hasSSN*))
Each object is uniquely identified by its social security number.

DataPropertyAssertion(*a:hasSSN a:Peter* "123-45-6789")
– Peter Griffin's social security number is "123-45-6789".

DataPropertyAssertion(*a:hasSSN a:Peter_Griffin* "123-45-6789")
Peter Griffin's social security number is "123-45-6789".

SameIndividual(*a:Peter a:Peter_Griffin*)
Peter and Peter Griffin are the same Person.

Carnegie Mellon

Knowledge elicitation through queries

SPARQL

Carnegie Mellon

Semantic query languages

- RDF is a W3C recommendation since 1998!
- Over the years, different approaches to query RDF repositories have been:
  - SQL-like: RDQL/Squish, SeRQL
  - Rule-based: N3QL, Triple, OWL-QL
  - XML-based: XSLT, XPath, Xquery
  - Xpathi-like: Versa, RDFPath
- SQL-like lead to the development of a new standard, SPARQL (10 years later).

SPARQL: Introduction

- SPARQL: Simple Protocol and RDF Query Language
- Syntactically similar to SQL (Structured Query Language) but different model
  - Relational DBs are characterized by tables, whose elements can be retrieved only by means of primary and foreign keys
  - In RDF knowledge bases, every element is identified by URI. RDF graphs can form a larger graph, graphs can be retrieved by URI matches

Differences between Primary and Foreign Keys

Primary Key	Foreign Key
Primary key uniquely identify a record in the table.	Foreign key is a field in the table that is primary key in another table.
Primary Key can't accept null values.	Foreign key can accept multiple null value.
By default, Primary key is clustered index and data in the database table is physically organized in the sequence of clustered index.	Foreign key do not automatically create an index, clustered or non-clustered. You can manually create an index on foreign key.
We can have only one Primary key in a table.	We can have more than one foreign key in a table.

Graph patterns

- SPARQL queries form a template called Graph Pattern
- Graph patterns need to match the RDF triples in the queried graph to return a result

```
PREFIX cra: http://cra.psu/edu/ontologies/CRATELO
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

SELECT ?x
   WHERE {?x rdfs:subClassOf cra:CyberAttack}
```

Carnegie Mellon

Querying OWL:

- SPARQL is made for RDF triples
- OWL is built on top of RDF
- OWL can be queried as a series of triples
  - Not very powerful (DL queries?)
  - Sneak peak (we'll see some examples soon):

```
SELECT ?restriction ?restrictionPredicate ?restrictionValue ?s
       WHERE {cra:Block-IP-attacker rdfs:subClassOf ?restriction.
              ?restriction owl:onProperty ?s.
              ?restriction ?restrictionPredicate ?restrictionValue}
```

Carnegie Mellon

More considerations on OWL

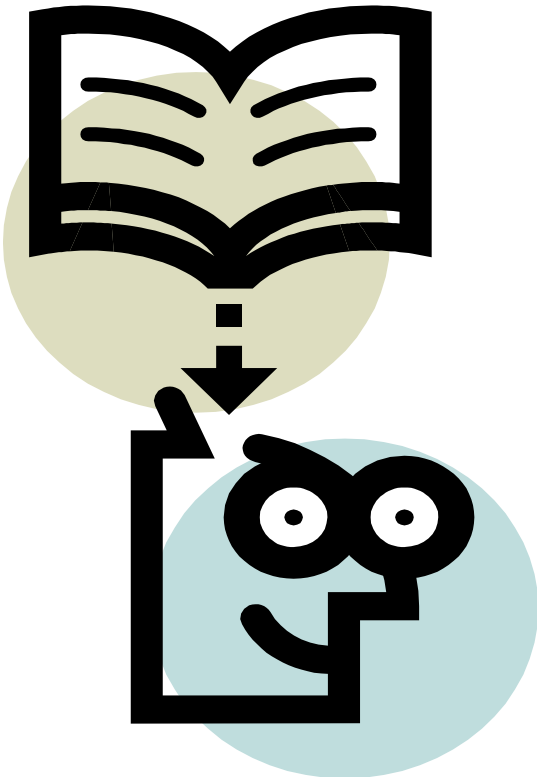
- Doesn't support temporal reasoning
  - …but you can import OWL-time.
- No probability
  - …but we Pr-OWL is a thing
- Needs more math operator (higher expressivity)
  - …spotty ad hoc solutions
- Shall we give up decidability?
  - Yes.

General references

- G. Antoniou and F. van Harmelen, “Web Ontology Language”, Handbook on Ontologies, Springer, 2004.

- M. Horridge, H. Knublauch, A. Rector, R. Stevens, C. Wroe, “A practical guide to building OWL ontologies using The Protégé-OWL plug-in and CO-ODE Tools Edition 1.0. Available at: http://protege.stanford.edu/

- http://protege.stanford.edu/plugins/owl/publications/2004-07-06-OWL-Tutorial.ppt
- http://www.w3.org/TR/owl-features/
- http://www.sts.tu-harburg.de/r.f.moeller/racer/
- http://www.mindswap.org/2003/pellet/
- http://www.cs.man.ac.uk/ horrocks/FaCT/
- http://www.w3.org/TR/2009/REC-owl2-overview-20091027/



Carnegie Mellon

Credits

- Leo Obrst, MITRE
- Armando Stellato, University of Rome, “Tor Vergata”
- James Handler, RPI.

Carnegie Mellon

OWL-2 Profiles

APPENDIX

Carnegie Mellon

Profiles

- OWL 1 defined two major dialects, OWL DL and OWL Full, and one syntactic subset (OWL Lite). However, it turned out that this was not sufficient to address requirements later identified by deployments of OWL ontologies.

- OWL 2 defined three different profiles according to different requirements:
  - OWL 2 EL
  - OWL 2 QL
  - OWL 2 RL

OWL 2 EL

- Many applications, particularly in the life sciences, use very large ontologies, e.g.; the FMA, NCI Thesaurus, SNOMED CT, Gene Ontology and some OBO ontologies. Such ontologies often need to represent (rather) complex entities (e.g.; anatomical entities composed of parts connected in complex ways) or to allow the propagation of properties (e.g.; location of diseases from parts to whole); they also have a huge number of classes, and heavy use is made of classification in order to facilitate development and maintenance. Applications are, therefore, mainly concerned with language scalability and reasoning performance problems

- OWL 2 EL captures the expressive power used by many large-scale ontologies, e.g.; SNOMED CT, and the NCI thesaurus.

- OWL 2 EL places several syntactic restrictions on the language

OWL 2 QL

- Many applications involving classical databases are concerned with interoperability of OWL with database technologies and tools. While the ontologies used in such applications are typically relatively lightweight, they are often used to query very large sets of individuals stored in standard relational databases. There is, therefore, a requirement to access such data directly via relational queries (e.g., SQL).

- OWL 2 QL captures the expressive power typically used in simple ontologies like thesauri, and (most of) the expressive power of ER/UML schemas.

- OWL 2 QL places several syntactic restrictions on the language.

OWL 2 RL

- Other applications are concerned with interoperability of the ontology language with rules and existing rule engines. While the ontologies used in such applications are again typically relatively lightweight, they may be used to query large datasets, and it may be useful or necessary to operate directly on data in the form of RDF triples. Typical cases include both OWL applications that are willing to trade the full expressivity of the language for efficiency, and RDF(S) applications that need some added expressivity from OWL 2.

- OWL 2 RL is designed to accommodate both OWL 2 applications that can trade the full expressivity of the language for efficiency, and RDF(S) applications that need some added expressivity from OWL 2. This is achieved by defining a syntactic subset of OWL 2 which is amenable to implementation using rule-based technologies.

- OWL 2 RL places several syntactic restrictions on the language

Which profile to choose?

The choice between the different profiles mainly depends on the expressiveness required by the application, the priority given to reasoning on classes or data, the size of datasets and importance of scalability, etc.

The following suggestions may be useful:

- Users requiring a scalable profile for large but (rather) simple ontologies and good time performance for ontology (TBox/schema) reasoning may want to consider OWL 2 EL.
- Users requiring a profile that can easily interoperate with relational database systems, and where scalable reasoning on large datasets is the most important task may want to consider OWL 2 QL.
- Users requiring a profile that can easily interoperate with rules engines, and where scalable reasoning on large datasets is the most important task may want to consider OWL 2 RL.