Combining Ontologies with Rules

(Two Different Worlds?)

Vassilis Papataxiarhis

Pervasive **C**omputing **R**esearch **G**roup
Communications **N**etwork **L**aboratory
Department Informatics and Telecommunications
University of Athens – Greece 2007

Outline

- Definition of ontologies and rules
- Integration Difficulties
- Integration Approaches
- Tools

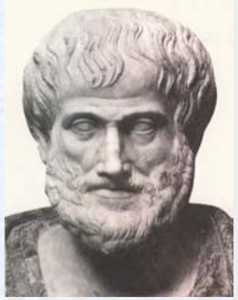
What really is an ontology? (1/2)

- Q: Is ontology an hierarchical structure of concepts?
- A: Yes, but not only that.

- Ontology=

  Όν (categories of being) +

  λόγος (treatise)



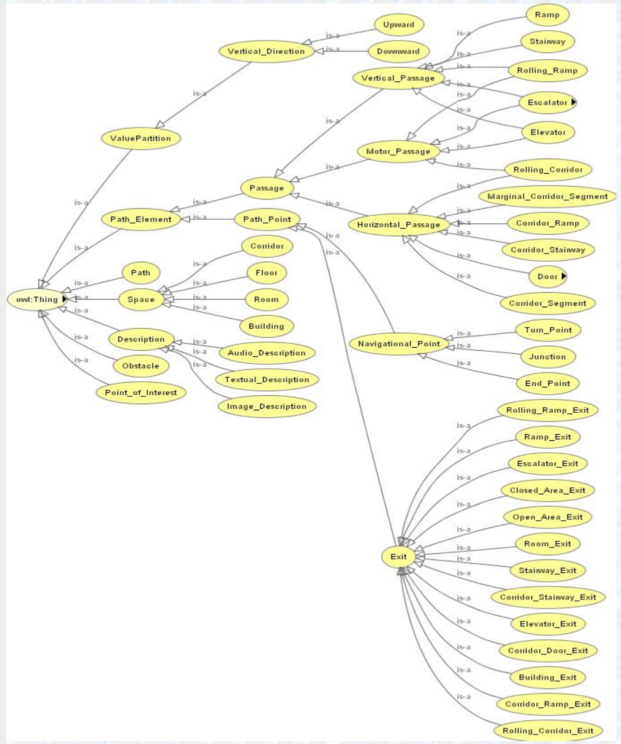
(i.e. the philosophy of being, *Metaphysics, Aristotle*).

- But in ancient greek λόγος = logic!

What really is an ontology? (2/2)

- Ontologies are used not only to represent a domain of interest, but also DEFINE concepts, describe relations among them and insert individuals.

- So, an ontology is not just a taxonomy like that

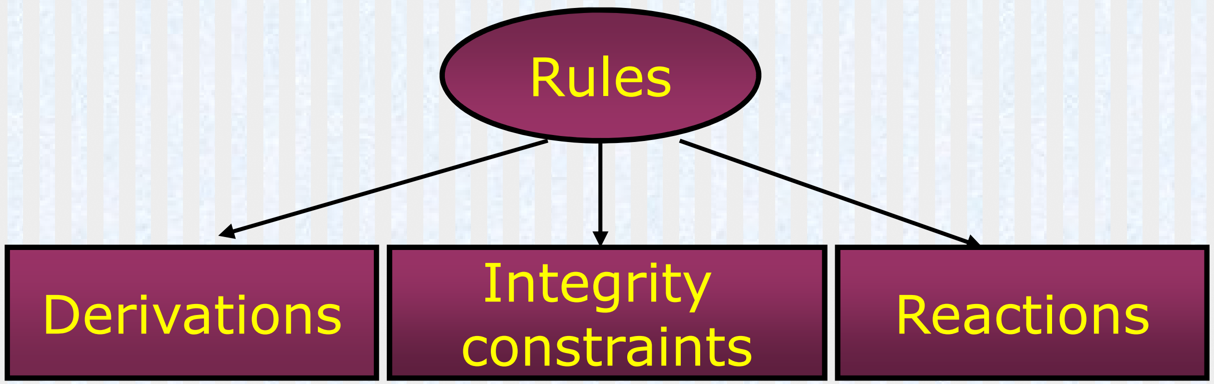


- Basic Ontology Languages:
  - Ontology Web Language (OWL)
  - DAML+OIL

- **Maturity**

Rules

- Rules are mainly based on subsets of First Order Logic (FOL) + possible extensions.
- Basic Rule Formalisms (in Semantic Web):
  - Semantic Web Rule Language (SWRL)
  - Answer Set Programming (ASP) (Datalog$^{\vee\neg}$)
- **Immaturity**

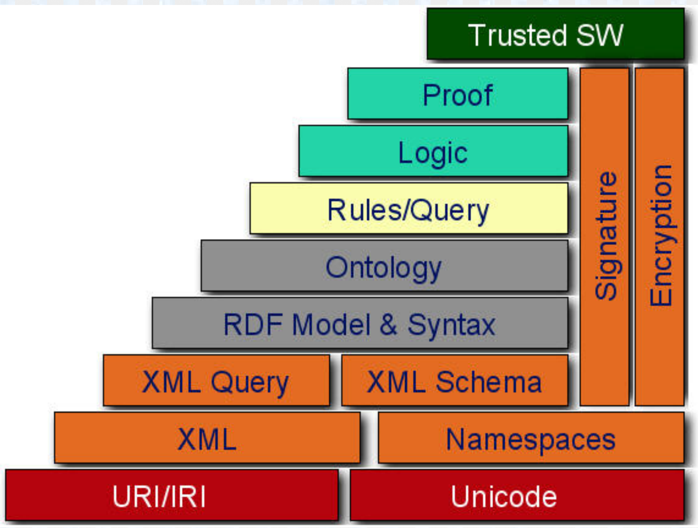


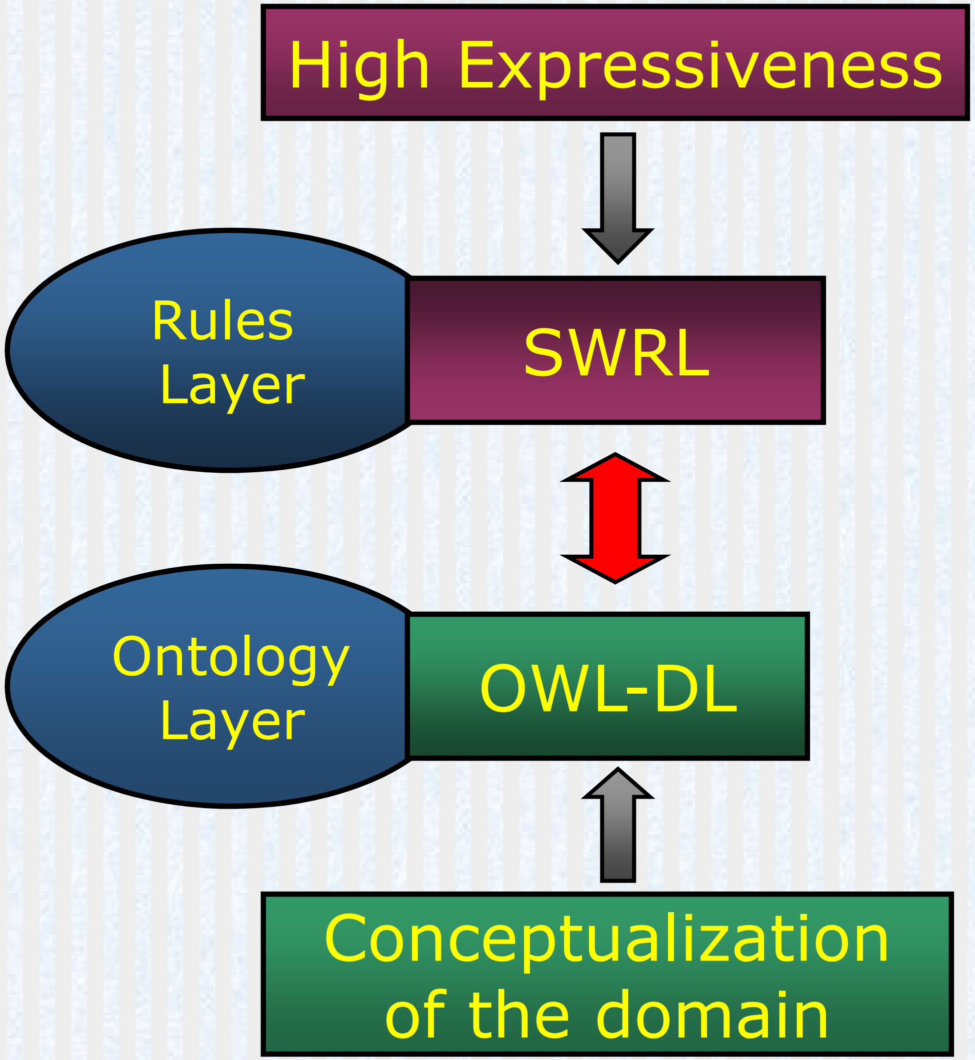
Why we need both of them?

- Ontologies are based on Description Logics (and thus in classical logic).
  - ✓ The Web is an open environment.
  - ✓ Reusability / interoperability.
  - ✓ An ontology is a model easy to understand.

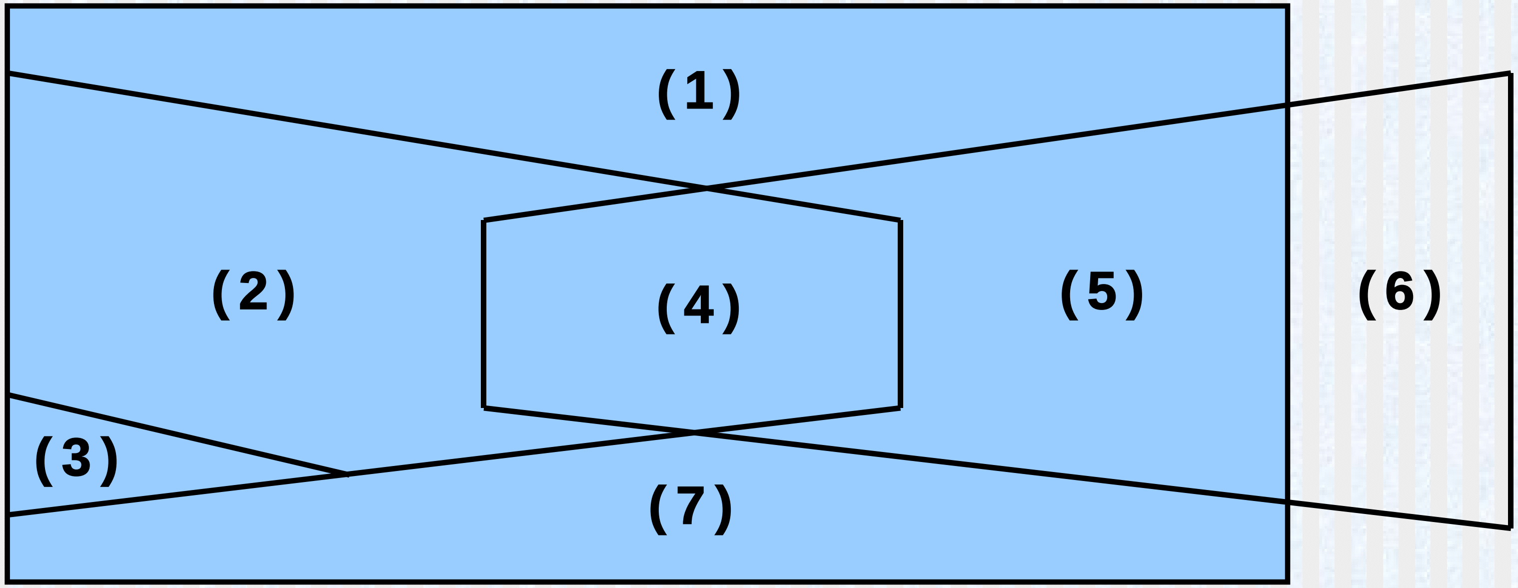
- Rules are based on logic programming.
  - ✓ For the sake of decidability, ontology languages don't offer the expressiveness we want (e.g. constructor for composite properties?). Rules do it well.
  - ✓ Efficient reasoning support already exists.
  - ✓ Rules are well-known in practice.

Usual combination





LP and Classical logic Overlap



FOL: (All except (6)), (2)+(3)+(4): DLs

(4): Description Logic Programs (DLP), (3): Classical Negation

(4)+(5): Horn Logic Programs, (4)+(5)+(6): LP

(6): Non-monotonic features (like NAF, etc.) (7): ^head and, vbody

Basic Difficulties

Classical Logic	vs.	Logic Programming

- Monotonic vs. Non-monotonic Features
  - Open-world vs. Closed-world assumption
  - Negation-as-failure vs. classical negation
- Non-ground entailment
- Strong negation vs. classical negation
- Equality
- Decidability

Open-world vs. Closed-world assumption

- Logic Programming – CWA
  - If KB $\not\models a$, then KB = KB $\cup$ $\neg a$
- Classical Logic – OWA
  - It keeps the world open.
  - KB:

    Man $\sqsubseteq$ Person, Woman $\sqsubseteq$ Person

    Bob $\in$ Man, Mary $\in$ Woman

  Query: “find all individuals that are not women”

Equality

- LP ----> Unique Name Assumption (UNA)
- Classical logic ----> different names may represent the same atom
- Example:

  differentPlayers(x,y) $\leftarrow$ *player(x), player(y),* $x \neq y$

  player(gerrard_of_liverpool).

  player(gerrard_of_england).

- In LP, we could conclude:

 $\neq$ 

  differentPlayers(gerrard_of_liverpool, gerrard_of_england)

Decidability

- The largest obstacle!
  - Tradeoff between expressiveness and decidability.
- Facing decidability issues from 2 different angles
  - In LP: Finiteness of the domain
  - In classical logic (and thus in DL): Combination of constructs
- Problem:

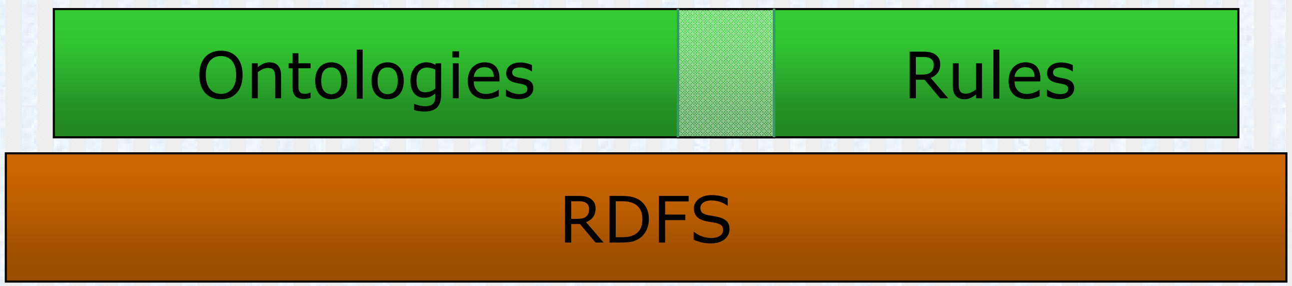
  Combination of “simple” DLs and Horn Logic are undecidable. (Levy & Rousset, 1998)

Rules + Ontologies

- Still a challenging task!
- A number of different approaches exists: SWRL, DLP (Grosof), dl-programs (Eiter), DL-safe rules, Conceptual Logic Programs (CLP), AL-Log, DL+log.
- 2 Main Strategies:
  - Tight Semantic Integration (Homogeneous Approaches)
  - Strict Semantic Separation (Hybrid Approaches)

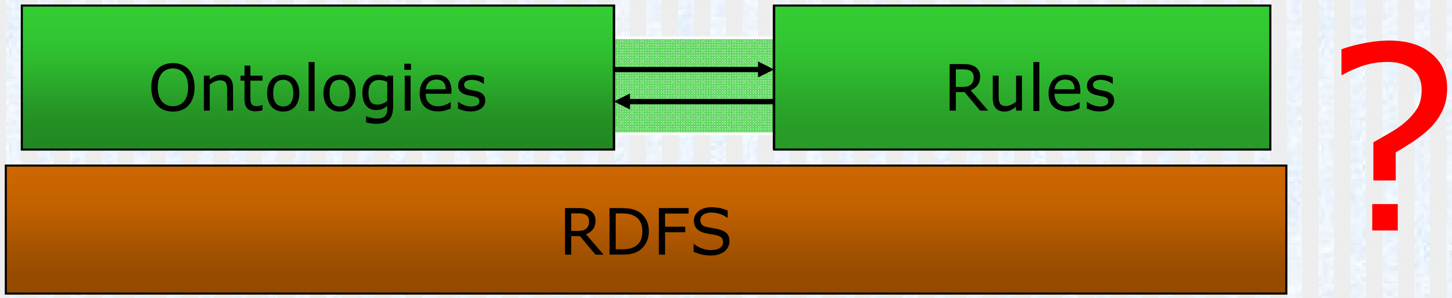
Homogeneous Approach

- Interaction with tight semantic integration.
- Both ontologies and rules are embedding in a common logical language.
- No distinction between rule predicates and ontology predicates.
- Rules may be used for defining classes and properties of the ontology.
- Example: SWRL, DLP



Hybrid Approach

- Integration with strict semantic separation between the two layers.
- Ontology is used as a conceptualization of the domain.
- Rules cannot define classes and properties of the ontology, but some application-specific relations.
- Communication via a “safe interface”.
- Example: Answer Set Programming (ASP)



SWRL

- Extend OWL axioms to include Horn-like clauses.
- Maximum compatibility with OWL
- Built on top of OWL (same semantics)
- Generic Formula:

$a_1 \wedge \ldots \wedge a_n \leftarrow b_1 \wedge . . . \wedge b_k$

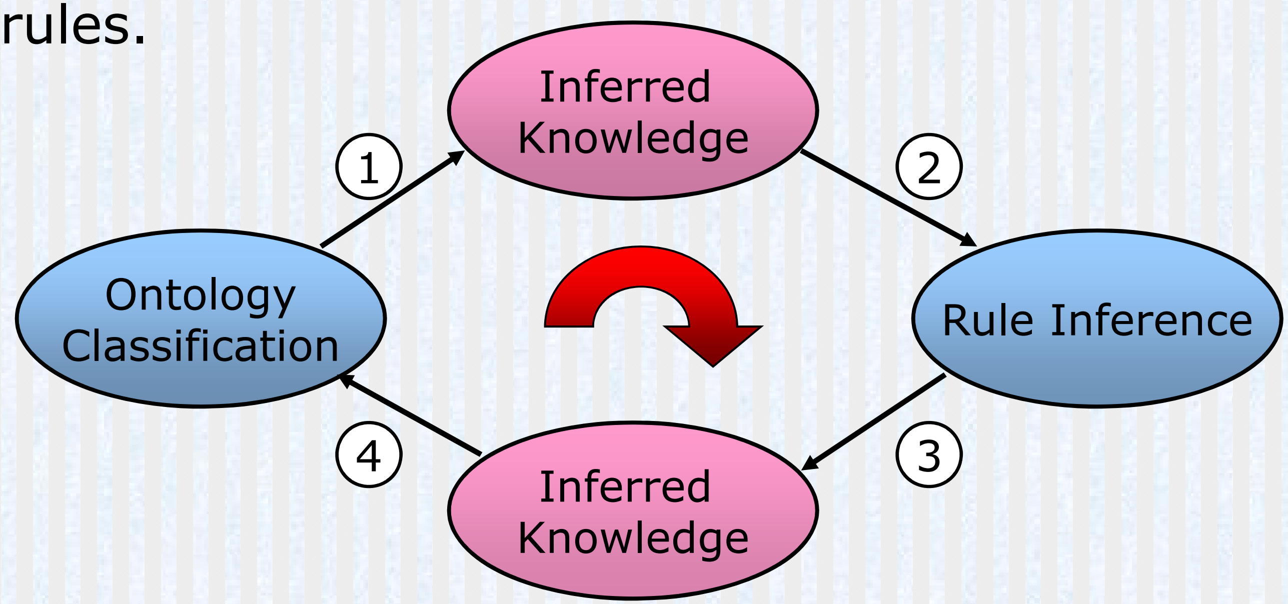
- Limitations
  - Negation, Disjunction
  - Undecidable

Tools

- Ontology Editors
  - Protégé, Swoop, TopBraid Composer
- Rule Editors
  - Protégé (SWRL-Tab)
- Ontology Reasoners
  - RacerPro, Bossam, Pellet, Fact++
- RuleEngines
  - Bossam, Jess, Jena Framework (only JRules)
  - ASP solvers: DLV, Smodels, nomore++

Limitations (1/2)

- The rule inference support is not integrated with an OWL classifier.
  - So, new assertions by rules may violate existing restrictions in ontology. New inferred knowledge from classification may in turn produce knowledge useful for rules.



Limitations (2/2)

- Existing solution:

  Solve these possible conflicts manually.
- Ideal solution:

  Have a single module for both ontology classification and rule inference.

- What if we want to combine non-monotonic features with classical logic?
  - Partial Solutions:
    - ASP
    - Externally (through the use of appropriate rule engines)