COMP318

Ontologies and Semantic Web

OWL - Part 2

Dr Valentina Tamma
V.Tamma@liverpool.ac.uk

Where were we

- RDF data modelling language & RDFS Schema language
- SPARQL query language
- Expressive limitations of RDFS as a schema language

Combining OWL with RDF Schema

- Ideally, OWL would extend RDF Schema
  - Consistent with the layered architecture of the Semantic Web
- Simply extending RDF schema would not give us expressive power and efficient reasoning at the same time:
  - combining RDF Schema with logics leads to uncontrollable computational properties...
- And there are layering issues:
  - Syntax
    - Only binary relations in RDF
    - Verbose Syntax
    - No limitations on graph in RDF
  - Semantics
    - Malformed graphs
    - Use of vocabulary in language, e.g. <rdfs:Class, rdfs:subClassOf, ex:a>
  - Meta-classes, e.g. ex:a, rdf:type, ex:a
- And what about the inference?

Extending RDFS

- OWL extends RDF Schema:
  - OWL is a KR language for the web
  - OWL Logical expressions (and, or, not)
    - `:Woman = :Human and :Female`
    - `:Person = :Man or :Woman`
    - `:Man = not (:Woman)`
  - **local properties**
    - `:Tenant only :rents :ResidentialUnits`
  - **(in)equality**
    - `:john differentFrom :mary`
    - `:JKRowling sameAs :RobertGalbraith`
- **required/optional properties**
  - `:rents :ResidentialUnit` is a required property for `:Tenant`
- **required values**
  - `:AmsterdamBuilding :hasLocation = :Amsterdam`
    - the value for the `:hasLocation` property for the class `:AmsterdamBuilding` can only be `:Amsterdam`
- **enumerated classes**
  - `DaysOfTheWeek = {Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday}`
- **symmetry, inverse and other property characteristics**

Reasoning support

- Formal semantics allows the automatic deduction of new facts and possible conflicts between class definitions (consistency):
  - `:Mammals` and `:Fish` are disjoint classes
    - i.e. they cannot have any common individuals
      - `:dolphin`?

- Automatic reasoning allows us to automatically performs checks that aim to detect the correctness of the model + instances (ontology).
  - These checks are extremely valuable for designing large ontologies, for collaborative ontology design and for sharing and integrating ontologies from various sources:
    - check the consistency of the ontology;
    - check for unintended relations between classes;
    - check for the unintended classification of instances.

Reasoning support

- Formal semantics allows the automatic deduction of new facts and possible conflicts between class definitions (consistency):

An ontology language can be provided with formal semantics and reasoning support by mapping it to a known logical formalism and by using the reasoning tools developed for the chosen formalism.

- These checks are extremely valuable for designing large ontologies, for collaborative ontology design and for sharing and integrating ontologies from various sources:
  - check the consistency of the ontology;
  - check for unintended relations between classes;
  - check for the unintended classification of instances.

OWL 1 and OWL 2

- OWL: OWL 1 (`http://www.w3.org/TR/owl-features/`) and OWL 2 (`http://www.w3.org/TR/owl2-overview/`)
  - Rationale for OWL
    - **Open world assumption:** the absence of a particular statement means that the statement has not been made explicitly yet.
      - Whether the statement is true or not, and whether it is believed that it is (or would be) true or not is irrelevant.
      - Thus, from the absence of a statement alone, a deductive reasoner cannot infer that the statement is false.
    - Reasonable trade-off between expressivity and scalability
    - Fully declarative semantics.
- **OWL DL**
  - Fragment of first order predicate logic, decidable
  - Known complexity classes
  - Reasonably efficient for real ontologies + instances

OWL 1: Three species of OWL

- OWL Lite:
  - Sublanguage of OWL DL but without nominals and XML datatypes:
  - Classification hierarchy
  - Simple constraints
  - It excludes enumerated classes, disjointness statements, and arbitrary cardinality.
  - Reasoning still not tractable.
- OWL DL
  - Sublanguage of OWL Full, it imposes restrictions on the use of OWL/RDFS constructors
  - Application of OWL's constructors to each other not permitted
  - Provides reasonably efficient reasoning support.

OWL 1: Three species of OWL

- OWL Full
  - Very high expressiveness, uses all of the OWL primitives
  - Fully upward compatible with RDF
  - Losing decidability: no complete or efficient reasoning support
  - All syntactic freedom of RDF (self-modifying):
    - primitives can be combined in arbitrary ways with RDF(S)

OWL 2 - Profile

- Sublanguages of OWL2 trading expressive power for efficient reasoning
  - Each supports different application scenarios
- OWL 2 EL
  - very large ontologies, efficient reasoning performance guaranteed at the expenses of expressive power;
- OWL 2 RL
  - subclass axioms understood as rule like implication, with head - superclass and body - subclass
  - different restrictions on subclasses and superclasses
  - allows the integration of OWL with rules

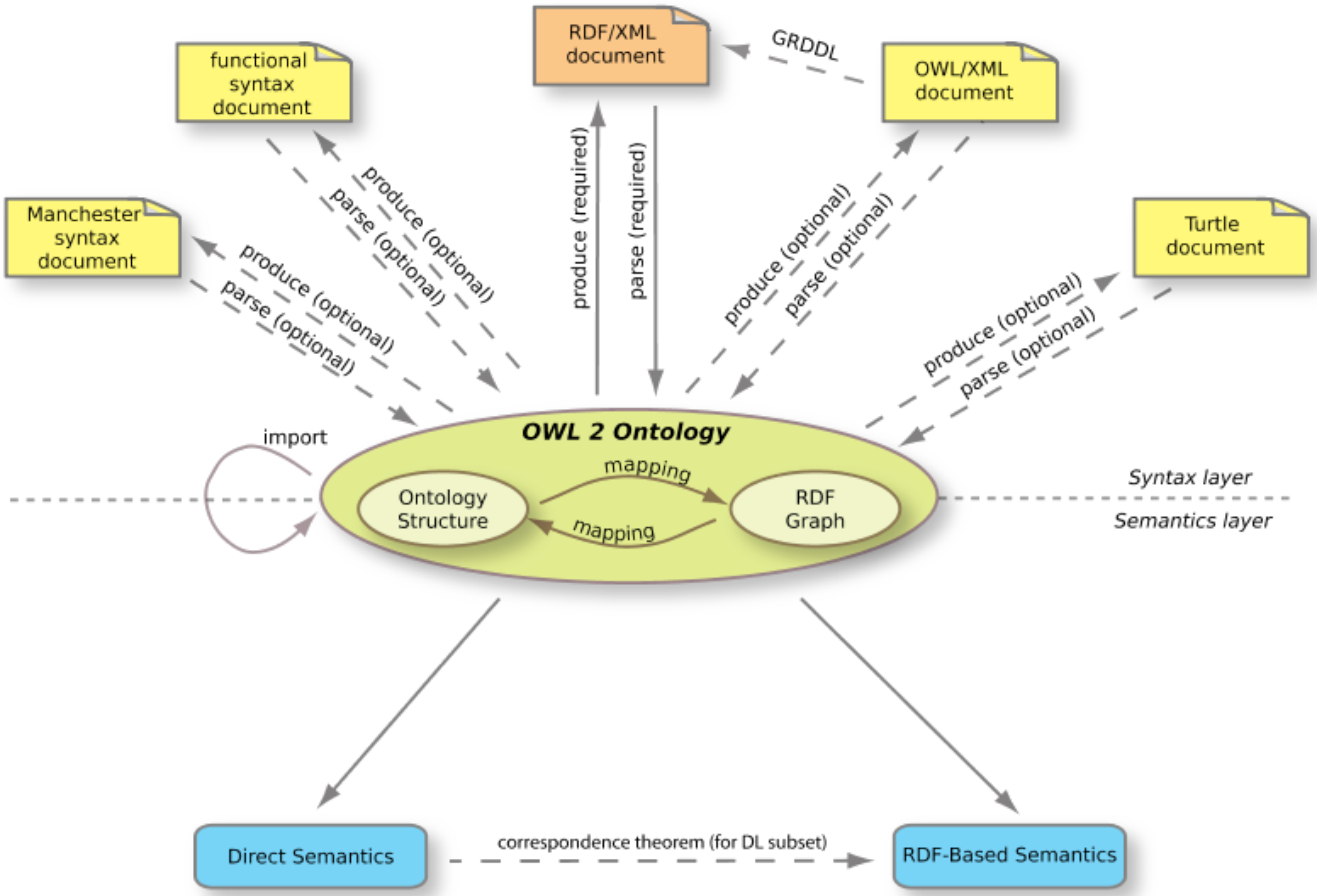
OWL 2 - Profile

- Sublanguages of OWL2 trading expressive power for efficient reasoning
  - Each supports different application scenarios
- OWL 2 QL
  - useful to query data rich applications
  - different restrictions on subclasses and superclasses
  - suitable for simple, lightweight ontologies with a large number of individuals and it is necessary to access the data directly via SQL queries
  - fast implementation on top of legacy DB systems, relational or RDF

OWL syntax

- RDF
  - Official exchange syntax
    - Hard for humans
    - RDF parsers are hard to write!
- UML
  - Large user base
- XML
  - Not the RDF syntax, it is not based on RDF conventions
    - Better for humans
    - More XML than RDF tools available
      - `http://www.w3.org/TR/owl-xmlsyntax/`
      - `http://www.w3.org/TR/owl2-mapping-to-rdf`
- Human readable syntax
  - Manchester syntax, used by Protege
  - Functional syntax, more compact and readable
    - `http://www.w3.org/2007/OWL/wiki/ManchesterSyntax`
    - `http://www.w3.org/TR/owl2-manchester-syntax/`
  - Turtle syntax for OWL 2
    - `http://www.w3.org/TR/owl2-primer/`

OWL 2 structure



COMP318

Ontologies and Semantic Web

End of OWL - Part 2

Dr Valentina Tamma
V.Tamma@liverpool.ac.uk