Semantic Web Technologies

Lecture 1: Web Ontology Language OWL

Maria Keet

email: keet -AT- inf.unibz.it

home: http://www.meteck.org

blog: http://keet.wordpress.com

KRDB Research Center
Free University of Bozen-Bolzano, Italy

16 November 2009

Outline

Limitations of RDFS

Web Ontology Language OWL
- Design of OWL
- OWL Layering
- OWL and Description Logics
- OWL Syntaxes

Layering OWL on top of RDF(S)

Slides based on Jos de Bruijn's slides of SWT '08/'09

Outline

Limitations of RDFS

Web Ontology Language OWL
- Design of OWL
- OWL Layering
- OWL and Description Logics
- OWL Syntaxes

Layering OWL on top of RDF(S)

RDFS as an Ontology Language

- Classes
- Properties
- Class hierarchies
- Property hierarchies
- Domain and range restrictions

Expressive limitations of RDF(S)

- Only binary relations
- Characteristics of Properties (e.g. inverse, transitive, symmetric)
- Local range restrictions (e.g. for Class Person, the property hasName has range xsd:string)
- Complex concept descriptions (e.g. Person is defined by Man and Woman)
- Cardinality restrictions (e.g. a Person may have at most 1 name)
- Disjointness axioms (e.g. nobody can be both a Man and a Woman)

Layering issues

- Syntax
  - Only binary relations in RDF
  - Verbose Syntax
  - No limitations on graph in RDF
    - Every graph is valid
- 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⟩

Outline

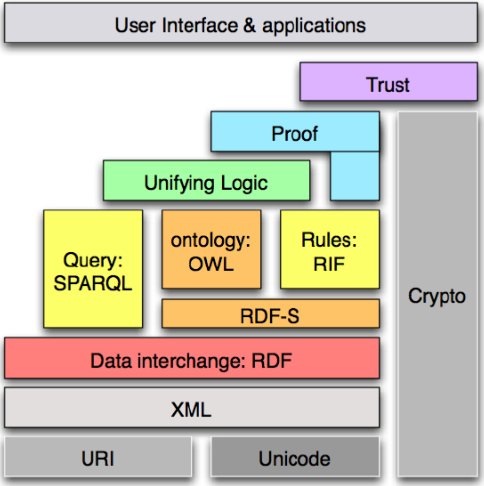
Limitations of RDFS

Web Ontology Language OWL

- Design of OWL
- OWL Layering
- OWL and Description Logics
- OWL Syntaxes

Layering OWL on top of RDF(S)

The place of OWL in the layer cake



Stack of Languages

- XML
  - Surface syntax, no semantics
- XML Schema
  - Describes structure of XML documents
- RDF
  - Datamodel for “relations” between “things”
- RDF Schema
  - RDF Vocabulary Definition Language
- OWL
  - A more expressive Vocabulary Definition Language

Design Goals for OWL

- **Shareable**
- **Changing** over time
- **Interoperability**
- **Inconsistency** detection
- Balancing **expressivity and complexity**
- **Ease of use**
- Compatible with **existing standards**
- **Internationalization**

Requirements for OWL

- Ontologies are **object on the Web**
- with **their own meta-data**, versioning, etc...
- Ontologies are **extendable**
- They contain **classes, properties, data-types, range/domain, individuals**
- **Equality** (for classes, for individuals)
- **Classes as instances**
- **Cardinality** constraints
- **XML** syntax

Objectives for OWL

Objectives:

- layered language
- complex datatypes
- digital signatures
- decidability (in part)
- local unique names (in part)

Disregarded:

- default values
- closed world option
- property chaining
- arithmetic
- string operations
- partial imports
- view definitions
- procedural attachments

Objectives for OWL

Objectives:

- layered language
- complex datatypes
- digital signatures
- decidability (in part)
- local unique names (in part)

Disregarded:

- default values
- closed world option
- property chaining
- arithmetic
- string operations
- partial imports
- view definitions
- procedural attachments

Objectives for OWL

Objectives:

- layered language
- complex datatypes
- digital signatures
- decidability (in part)
- local unique names (in part)

Disregarded:

- default values
- closed world option
- property chaining
- arithmetic
- string operations
- partial imports
- view definitions
- procedural attachments

Extending RDF Schema

- Leveraging experiences with OWL's predecessors SHOE, OIL, DAML-ONT, and DAML+OIL (frames, OO, DL)
- OWL extends RDF Schema to a full-fledged knowledge representation language for the Web
  - Logical expressions (and, or, not)
  - (in)equality
  - local properties
  - required/optional properties
  - required values
  - enumerated classes
  - symmetry, inverse

Species of OWL

- OWL Lite
  - Classification hierarchy
  - Simple constraints
- OWL DL
  - Maximal expressiveness
  - While maintaining tractability
  - Standard formalization in a DL
- OWL Full
  - Very high expressiveness
  - Losing tractability
  - All syntactic freedom of RDF (self-modifying)

Species of OWL

- OWL Lite
  - Classification hierarchy
  - Simple constraints
- OWL DL
  - Maximal expressiveness
  - While maintaining tractability
  - Standard formalization in a DL
- OWL Full
  - Very high expressiveness
  - Losing tractability
  - All syntactic freedom of RDF (self-modifying)

Species of OWL

- OWL Lite
  - Classification hierarchy
  - Simple constraints
- OWL DL
  - Maximal expressiveness
  - While maintaining tractability
  - Standard formalization in a DL
- OWL Full
  - Very high expressiveness
  - Losing tractability
  - All syntactic freedom of RDF (self-modifying)

Species of OWL

- OWL Lite
  - Classification hierarchy
  - Simple constraints
- OWL DL
  - Maximal expressiveness
  - While maintaining tractability
  - Standard formalization in a DL
- OWL Full
  - Very high expressiveness
  - Losing tractability
  - All syntactic freedom of RDF (self-modifying)

Features of OWL languages

- OWL Lite
  - (sub)classes, individuals
  - (sub)properties, domain, range
  - conjunction
  - (in)equality
  - (unqualified) cardinality 0/1
  - datatypes
  - inverse, transitive, symmetric properties
  - someValuesFrom
  - allValuesFrom
- OWL DL
  - Negation
  - Disjunction
  - (unqualified) Full cardinality
  - Enumerated classes
  - hasValue
- OWL Full
  - Meta-classes
  - Modify language

Features of OWL languages

- OWL Lite
  - (sub)classes, individuals
  - (sub)properties, domain, range
  - conjunction
  - (in)equality
  - (unqualified) cardinality 0/1
  - datatypes
  - inverse, transitive, symmetric properties
  - someValuesFrom
  - allValuesFrom
- OWL DL
  - Negation
  - Disjunction
  - (unqualified) Full cardinality
  - Enumerated classes
  - hasValue
- OWL Full
  - Meta-classes
  - Modify language

Features of OWL languages

- OWL Lite
  - (sub)classes, individuals
  - (sub)properties, domain, range
  - conjunction
  - (in)equality
  - (unqualified) cardinality 0/1
  - datatypes
  - inverse, transitive, symmetric properties
  - someValuesFrom
  - allValuesFrom
- OWL DL
  - Negation
  - Disjunction
  - (unqualified) Full cardinality
  - Enumerated classes
  - hasValue
- OWL Full
  - Meta-classes
  - Modify language

Features of OWL languages

- OWL Lite
  - (sub)classes, individuals
  - (sub)properties, domain, range
  - conjunction
  - (in)equality
  - (unqualified) cardinality 0/1
  - datatypes
  - inverse, transitive, symmetric properties
  - someValuesFrom
  - allValuesFrom
- OWL DL
  - Negation
  - Disjunction
  - (unqualified) Full cardinality
  - Enumerated classes
  - hasValue
- OWL Full
  - Meta-classes
  - Modify language

OWL Full

- **No restriction on use of vocabulary** (as long as legal RDF)
  - Classes as instances (and much more)
- **RDF style model theory**
  - Reasoning using FOL engine
  - Semantics should correspond to OWL DL for restricted KBs

OWL DL

- Use of vocabulary restricted
  - Cannot be used to do "nasty things" (e.g., modify OWL)
  - No classes as instances (this will be discussed in a later lecture)
  - Defined by abstract syntax
- Standard DL-based model theory
  - Direct correspondence with a DL
  - Automated reasoning with DL reasoners (e.g., Racer, Pellet, FaCT^{++})

OWL Lite

- No explicit negation or union
- Restricted cardinality (0/1)
- No nominals (oneOf)
- DL-based semantics
  - Automated reasoning with DL reasoners (e.g., Racer, Pellet, FaCT^{++})

More on OWL species

- OWL Full is *not* a Description Logic
- OWL Lite has strong syntactic restrictions, but only limited semantics restrictions cf. OWL DL
  - Negation can be encoded using disjointness
  - With negation an conjunction, you can encode disjunction
- For instance:

```
Class(C complete unionOf(B C))
```

is equivalent to:

```
DisjointClasses(notB B)
DisjointClasses(notC C)
Class(notBandnotC complete notB notC)
DisjointClasses(notBandnotC BorC)
Class(C complete notBandnotC)
```

More on OWL species

- OWL Full is *not* a Description Logic
- OWL Lite has strong syntactic restrictions, but only limited semantics restrictions cf. OWL DL
  - Negation can be encoded using disjointness
  - With negation an conjunction, you can encode disjunction
- For instance:

```
Class(C complete unionOf(B C))
```

is equivalent to:

```
DisjointClasses(notB B)
DisjointClasses(notC C)
Class(notBandnotC complete notB notC)
DisjointClasses(notBandnotC BorC)
Class(C complete notBandnotC)
```

More on layering and OWL flavours

- For an OWL DL-restricted KB, OWL Full semantics is **not** equivalent to OWL DL semantics

John friend Susan .

OWL Full entails:

John rdf:type owl:Thing . Susan rdf:type owl:Thing . friend rdf:type owl:ObjectProperty .

John rdf:type _:x . _:x owl:onProperty friend . _:x owl:minCardinality "1"^^xsd:nonNegativeInteger .

OWL and Description Logics

- OWL Lite corresponds to the DL $\mathcal{SHIF}(\mathbf{D})$
  - Named classes (A)
  - Named properties (P)
  - Individuals ($C(o)$)
  - Property values ($P(o, a)$)
  - Intersection ($C \sqcap D$)
  - Union ($C \sqcup D$)
  - Negation ($\neg C$)
  - Existential value restrictions ($\exists P.C$)
  - Universal value restrictions ($\forall P.C$)
  - Unqualified (0/1) number restrictions ($\geq nP$, $\leq nP$, $= nP$), $0 \leq n \leq 1$
- OWL DL corresponds to the DL $\mathcal{SHOIN}(\mathbf{D})$
  - Arbitrary number restrictions ($\geq nP$, $\leq nP$, $= nP$), $0 \leq n$
  - Property value ($\exists P.\{o\}$)
  - Enumeration ($\{o_1, ..., o_n\}$)

OWL and Description Logics

- OWL Lite corresponds to the DL $\mathcal{SHIF}(\mathbf{D})$
  - Named classes (A)
  - Named properties (P)
  - Individuals ($C(o)$)
  - Property values ($P(o, a)$)
  - Intersection ($C \sqcap D$)
  - Union ($C \sqcup D$)
  - Negation ($\neg C$)
  - Existential value restrictions ($\exists P.C$)
  - Universal value restrictions ($\forall P.C$)
  - Unqualified (0/1) number restrictions ($\geq nP$, $\leq nP$, $= nP$), $0 \leq n \leq 1$
- OWL DL corresponds to the DL $\mathcal{SHOIN}(\mathbf{D})$
  - Arbitrary number restrictions ($\geq nP$, $\leq nP$, $= nP$), $0 \leq n$
  - Property value ($\exists P.\{o\}$)
  - Enumeration ($\{o_1, ..., o_n\}$)

OWL and Description Logics

- OWL Lite corresponds to the DL $\mathcal{SHIF}(\mathbf{D})$
  - Named classes (A)
  - Named properties (P)
  - Individuals ($C(o)$)
  - Property values ($P(o, a)$)
  - Intersection ($C \sqcap D$)
  - Union ($C \sqcup D$)
  - Negation ($\neg C$)
  - Existential value restrictions ($\exists P.C$)
  - Universal value restrictions ($\forall P.C$)
  - Unqualified (0/1) number restrictions ($\geq nP$, $\leq nP$, $= nP$), $0 \leq n \leq 1$
- OWL DL corresponds to the DL $\mathcal{SHOIN}(\mathbf{D})$
  - Arbitrary number restrictions ($\geq nP$, $\leq nP$, $= nP$), $0 \leq n$
  - Property value ($\exists P.\{o\}$)
  - Enumeration ($\{o_1, ..., o_n\}$)

OWL constructs

OWL Construct	DL	Example
intersectionOf	$C_1 \sqcap \ldots \sqcap C_n$	$Human \sqcap Male$
unionOf	$C_1 \sqcup \ldots \sqcup C_n$	$Doctor \sqcup Lawyer$
complementOf	$\neg C$	$\neg Male$
oneOf	$\{o_1, \ldots, o_n\}$	$\{john, mary\}$
allValuesFrom	$\forall P.C$	$\forall hasChild.Doctor$
someValuesFrom	$\exists P.C$	$\exists hasChild.Lawyer$
value	$\exists P.\{o\}$	$\exists citizenOf.USA$
minCardinality	$\geq nP.C$	$\geq 2 hasChild.Lawyer$
maxCardinality	$\leq nP.C$	$\leq 1 hasChild.Male$
cardinality	$= nP.C$	$= 1 hasParent.Female$

+ XML Schema datatypes: int, string, real, etc...

OWL axioms

OWL Axiom	DL	Example
SubClassOf	$C_1 \sqsubseteq C_2$	$Human \sqsubseteq Animal \sqcap Biped$
EquivalentClasses	$C_1 \equiv \ldots \equiv C_n$	$Man \equiv Human \sqcap Male$
SubPropertyOf	$P_1 \sqsubseteq P_2$	$hasDaughter \sqsubseteq hasChild$
EquivalentProperties	$P_1 \equiv \ldots \equiv P_n$	$cost \equiv price$
SameIndividual	$o_1 = \ldots = o_n$	$President\_Bush = G\_W\_Bush$
DisjointClasses	$C_i \sqsubseteq \neg C_j$	$Male \sqsubseteq \neg Female$
DifferentIndividuals	$o_i \neq o_j$	$john \neq peter$
inverseOf	$P_1 \equiv P_2^-$	$hasChild \equiv hasParent^-$
Transitive	$P^+ \sqsubseteq P$	$ancestor^+ \sqsubseteq ancestor$
Symmetric	$P \equiv P^-$	$connectedTo \equiv connectedTo^-$

DL-based OWL species as Semantic Web languages vs DLs

- ⇒ OWL uses URI references as names (like used in RDF, e.g., `http://www.w3.org/2002/07/owl#Thing`)
- ⇒ OWL gathers information into ontologies stored as documents written in RDF/XML, things like `owl:imports`
- ⇒ RDF data types and XML schema data types for the ranges of data properties (attributes) (`DataPropertyRange`)
- OWL-DL and OWL-Lite with a frame-like abstract syntax, whereas RDF/XML is the official exchange syntax for OWL
- Annotations
- Note: DLs will receive full attention in the "Knowledge Representation and Ontologies" course in the next semester

Syntaxes of OWL

- RDF
  - Official exchange syntax
  - Hard for humans
  - RDF parsers are hard to write!
- XML
  - Not the RDF syntax
  - Still hard for humans, but more XML than RDF tools available
- Abstract syntax
  - Not defined for OWL Full
  - To some, considered human readable
- User-usable ones
  - e.g., Manchester syntax, informal and limited matching with UML

OWL in RDF/XML

Example from [OwlGuide]:

```
<!ENTITY vin
"http://www.w3.org/TR/2004/REC-owl-guide-20040210/wine#" >
<!ENTITY food
"http://www.w3.org/TR/2004/REC-owl-guide-20040210/food#" > ...
<rdf:RDF
xmlns:vin="http://www.w3.org/TR/2004/REC-owl-guide-20040210/wine#"
xmlns:food="http://www.w3.org/TR/2004/REC-owl-guide-20040210/food#"
 ... >

<owl:Class rdf:ID="Wine"> <rdfs:subClassOf
rdf:resource="&food;PotableLiquid"/> <rdfs:label
xml:lang="en">wine</rdfs:label> <rdfs:label
xml:lang="fr">vin</rdfs:label> ...  </owl:Class>

<owl:Class rdf:ID="Pasta"> <rdfs:subClassOf
rdf:resource="#EdibleThing" /> ... </owl:Class> </rdf:RDF>
```

OWL Abstract syntax

Class(professor partial) Class(associateProfessor partial academicStaffMember)

DisjointClasses (associateProfessor assistantProfessor)
DisjointClasses (professor associateProfessor)

Class(faculty complete academicStaffMember)

OWL Abstract syntax

In DL syntax:

$\text{associateProfessor} \sqsubseteq \text{academicStaffMember}$
$\text{associateProfessor} \sqsubseteq \neg\ \text{assistantProfessor}$
$\text{professor} \sqsubseteq \neg\ \text{associateProfessor}$
$\text{faculty} \equiv \text{academicStaffMember}$

More examples

```
DatatypeProperty(age range(xsd:nonNegativeInteger))
ObjectProperty( lecturesIn )

ObjectProperty(isTaughtBy domain(course) range(academicStaffMember))
SubPropertyOf(isTaughtBy involves)

ObjectProperty(teaches  inverseOf(isTaughtBy)
domain(academicStaffMember) range(course))

 EquivalentProperties ( lecturesIn   teaches)

ObjectProperty(hasSameGradeAs Transitive Symmetric domain(student)
range(student))
```

More examples

In DL syntax:

$\top \sqsubseteq \forall age.xsd : nonNegativeInteger$
$\top \sqsubseteq \forall isTaughtBy^{-}.course$
$\top \sqsubseteq \forall isTaughtBy.academicStaffMember$
$isTaughtBy \sqsubseteq involves$
$teaches \equiv isTaughtBy^{-}$
$\top \sqsubseteq \forall teaches^{-}.academicStaffMember$
$\top \sqsubseteq \forall teaches.course$
$lecturesIn \equiv teaches$
$hasSameGradeAs^{+} \sqsubseteq hasSameGradeAs$
$hasSameGradeAs \equiv hasSameGradeAs^{-}$
$\top \sqsubseteq \forall hasSameGradeAs^{-}.student$
$\top \sqsubseteq \forall hasSameGradeAs.student$

More examples

Individual (949318 type(lecturer))

Individual (949352 type(academicStaffMember) value(age "39"^^&xsd;integer))

ObjectProperty(isTaughtBy Functional)

Individual (CIT1111 type(course) value(isTaughtBy 949352) value(isTaughtBy 949318))

DifferentIndividuals (949318 949352) DifferentIndividuals (949352 949111 949318)

More examples

In DL syntax:

$949318 : lecturer$
$949352 : academicStaffMember$
$\langle 949352, "39"\hat{}\hat{}\&xsd;integer \rangle : age$
$\top \sqsubseteq \leq 1 isTaughtBy$
$CIT1111 : course$
$\langle CIT1111, 949352 \rangle : isTaughtBy$
$\langle CIT1111, 949318 \rangle : isTaughtBy$
$949318 \neq 949352$
$949352 \neq 949111$
$949111 \neq 949318$
$949352 \neq 949318$

More examples

Class(firstYearCourse partial restriction (isTaughtBy allValuesFrom (Professor)))

Class(mathCourse partial restriction (isTaughtBy hasValue (949352)))

Class(academicStaffMember partial restriction (teaches someValuesFrom (undergraduateCourse)))

Class(course partial restriction (isTaughtBy minCardinality (1)))

Class(department partial restriction (hasMember minCardinality(10)) restriction (hasMember maxCardinality(30)))

More examples

In DL syntax:

$$firstYearCourse \sqsubseteq \forall isTaughtBy.Professor$$
$$mathCourse \sqsubseteq \exists isTaughtBy.\{949352\}$$
$$academicStaffMember \sqsubseteq \exists teaches.undergraduateCourse$$
$$course \sqsubseteq \geq 1 isTaughtBy$$
$$department \sqsubseteq \geq 10 hasMember \sqcap \leq 30 hasMember$$

More examples

Class(course partial complementOf(staffMember))

Class(peopleAtUni complete unionOf(staffMember student))

Class(facultyInCS complete intersectionOf(faculty
restriction (belongsTo hasValue (CSDepartment))))

Class(adminStaff complete intersectionOf(staffMember
complementOf(unionOf(faculty techSupportStaff))))

More examples

In DL syntax:

$$course \sqsubseteq \neg staffMember$$
$$peopleAtUni \equiv staffMember \sqcup student$$
$$facultyInCS \equiv faculty \sqcap \exists belongsTo.\{CSDepartment\}$$
$$adminStaff \equiv staffMember \sqcap \neg(faculty \sqcup techSupportStaff)$$

Outline

Limitations of RDFS

Web Ontology Language OWL
- Design of OWL
- OWL Layering
- OWL and Description Logics
- OWL Syntaxes

Layering OWL on top of RDF(S)

Layering on top of RDF(S)

- RDF(S) bottom layer in Semantic Web stack
- Higher languages *layer* on top of RDFS

Syntactic Layering

- *Every valid RDF statement is a valid statement in a higher language*
- This *includes* triples containing keywords of these languages(!)

Semantic Layering

For RDFS graph G and higher-level language L:

If $G \models_{RDFS} G'$ then $G \models_L G'$, and *ideally*

if $G \models_L G'$ then $G \models_{RDFS} G'$

Syntactically layering OWL on RDF(S)

OWL Lite, OWL DL

- OWL Lite, OWL DL subsets of RDF
- Allowed triples defined through mapping from abstract syntax
- *Partial* layering:
  - *every* OWL Lite/DL ontology is an RDF graph
  - *some* RDF graphs are OWL Lite/DL ontologies

OWL Full

- OWL Full encompasses RDF
- *Complete* layering:
  - *every* OWL Full is an RDF graph
  - *all* RDF graphs are OWL Full ontologies

Semantically layering OWL on RDF(S)

OWL Lite, OWL DL

- OWL Lite/DL semantics *not* related to RDFS semantics
- Redefine semantics of RDFS keywords, e.g., `rdfs:subClassOf`
- Work ongoing to describe correspondence between subset of RDFS and OWL Lite/DL

OWL Full

- OWL Full semantics is *extension* of RDFS semantics
- OWL Full is undecidable
- OWL Full semantics hard to understand

OWL Lite/DL vs. RDF

- RDF Graph defined through translation from Abstract Syntax
- Example:

```
Class(Human partial Animal
            restriction(hasLegs cardinality(2))
            restriction(hasName allValuesFrom(xsd:string)))
```

Human	rdf:type	owl:Class
Human	rdfs:subClassOf	Animal
Human	rdfs:subClassOf	_:X1
_:X1	rdf:type	owl:Restriction
_:X1	owl:onProperty	hasLegs
_:X1	owl:cardinality	"2"^^xsd:nonNegativeInteger
Human	rdfs:subClassOf	_:X2
_:X2	rdf:type	owl:Restriction
_:X2	owl:onProperty	hasName
_:X2	owl:allValuesFrom	xsd:string

OWL Lite/DL vs. RDF

- RDF Graph defined through translation from Abstract Syntax
- Example:

```
Class(Human partial Animal
        restriction(hasLegs cardinality(2))
        restriction(hasName allValuesFrom(xsd:string)))
```

Human	rdf:type	owl:Class
Human	rdfs:subClassOf	Animal
Human	rdfs:subClassOf	_:X1
_:X1	rdf:type	owl:Restriction
_:X1	owl:onProperty	hasLegs
_:X1	owl:cardinality	"2"8sd:nonNegativeInteger
Human	rdfs:subClassOf	_:X2
_:X2	rdf:type	owl:Restriction
_:X2	owl:onProperty	hasName
_:X2	owl:allValuesFrom	xsd:string

OWL Lite/DL vs. RDF

- Not every RDF graph is OWL Lite/DL ontology
- Example:

  A rdf:type A

- How to check whether an RDF graph G is OWL DL?
  1. Construct an OWL ontology O in Abstract Syntax
  2. Translate to RDF graph G'
  3. If G=G', then G is OWL DL
     - Otherwise, go to step (1)

OWL Lite/DL vs. RDF

- Not every RDF graph is OWL Lite/DL ontology
- Example:

  A rdf:type A

- How to check whether an RDF graph G is OWL DL?
  1. Construct an OWL ontology O in Abstract Syntax
  2. Translate to RDF graph G'
  3. If $G = G'$, then G is OWL DL
     - Otherwise, go to step (1)

Summary

Limitations of RDFS

Web Ontology Language OWL
- Design of OWL
- OWL Layering
- OWL and Description Logics
- OWL Syntaxes

Layering OWL on top of RDF(S)

The future of OWL... is now

- Section 8 of Horrocks *et. al.*'s paper outlines possible "Future extensions"
- OWL 2 has become a W3C recommendation on 27 Oct 2009
- We look at the new recommendation in the following lectures

The future of OWL... is now

- Section 8 of Horrocks *et. al.*'s paper outlines possible "Future extensions"
- OWL 2 has become a W3C recommendation on 27 Oct 2009
- We look at the new recommendation in the following lectures