

# The OWL API

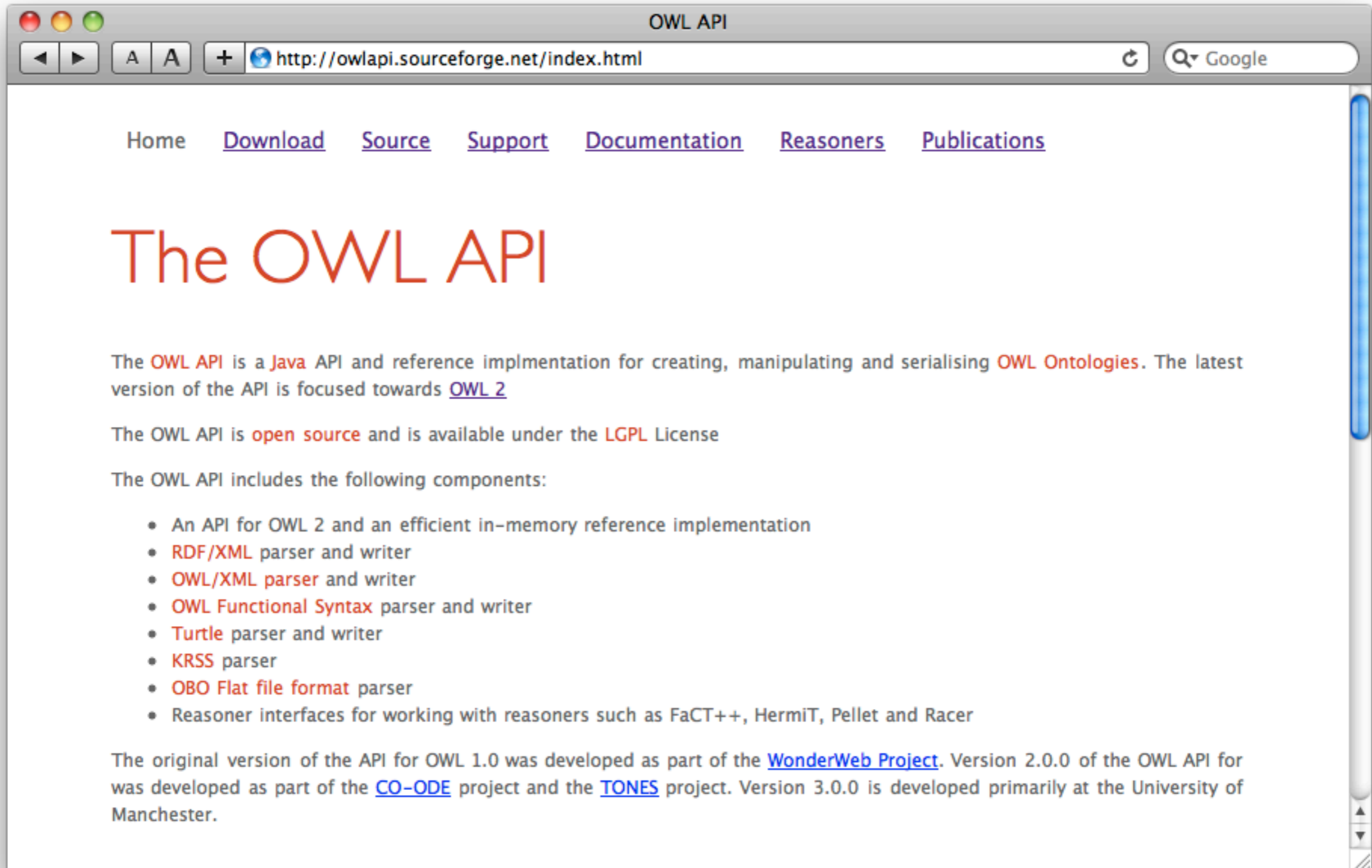
## A Java API for Working with OWL 2 Ontologies

Matthew Horridge

with special thanks to

Sean Bechhofer, Ron Alford, Nick Drummond, Birte Glimm, Olaf Noppens,  
Ignazio Palmisano, Timothy Redmond, Thomas Schneider,  
Evren Sirin, and Mike Smith

**`http://owlapi.sourceforge.net`**



# The OWL API

2003

Version 1

OWL 1



2007

Version 2

Interim OWL 2  
(OWL 1.1)



2009

Version 3

OWL 2

High level syntax neutral  
interfaces based on  
the OWL Abstract Syntax

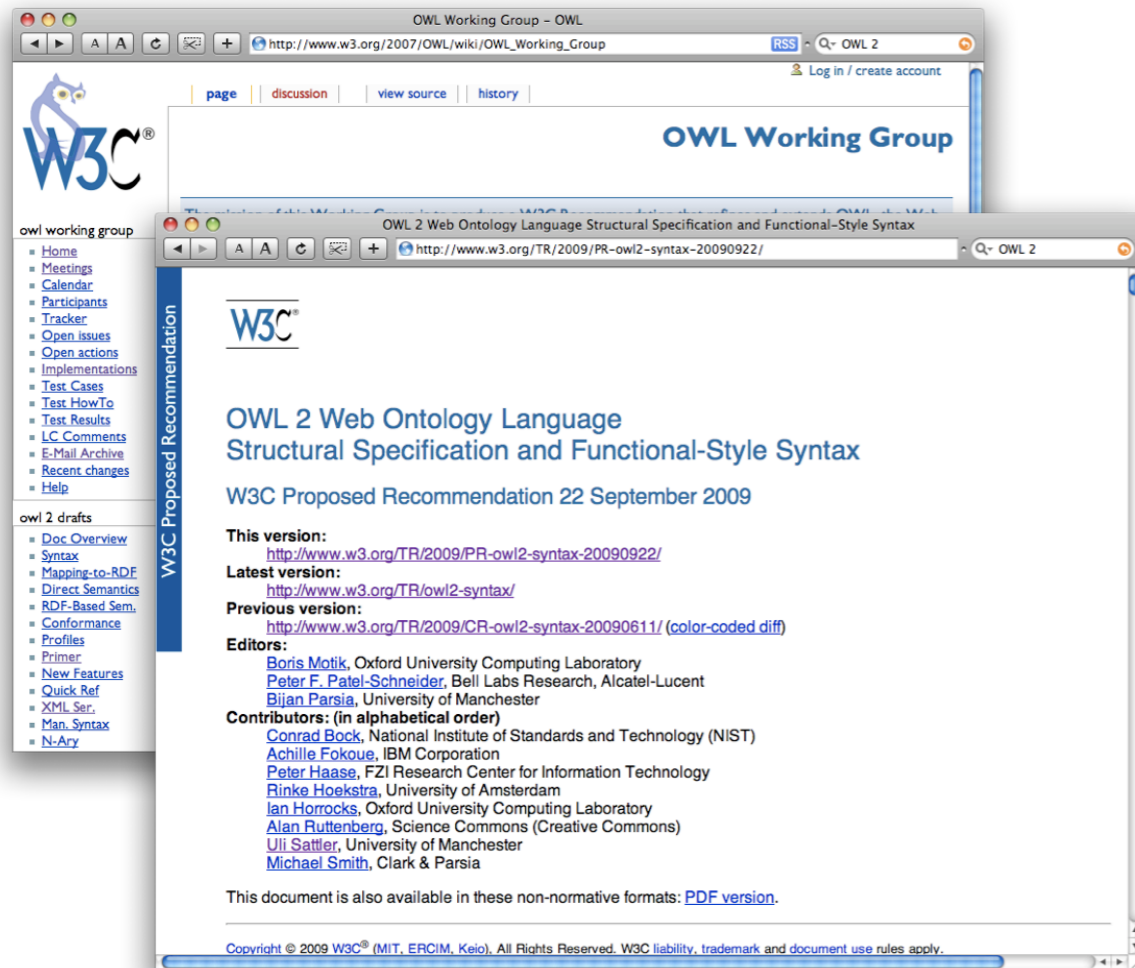
All changes applied through  
change objects

Shift from “frame”  
oriented model  
to axiom oriented model

Close alignment with  
OWL 2 specification

Addition of more  
convenience functionality

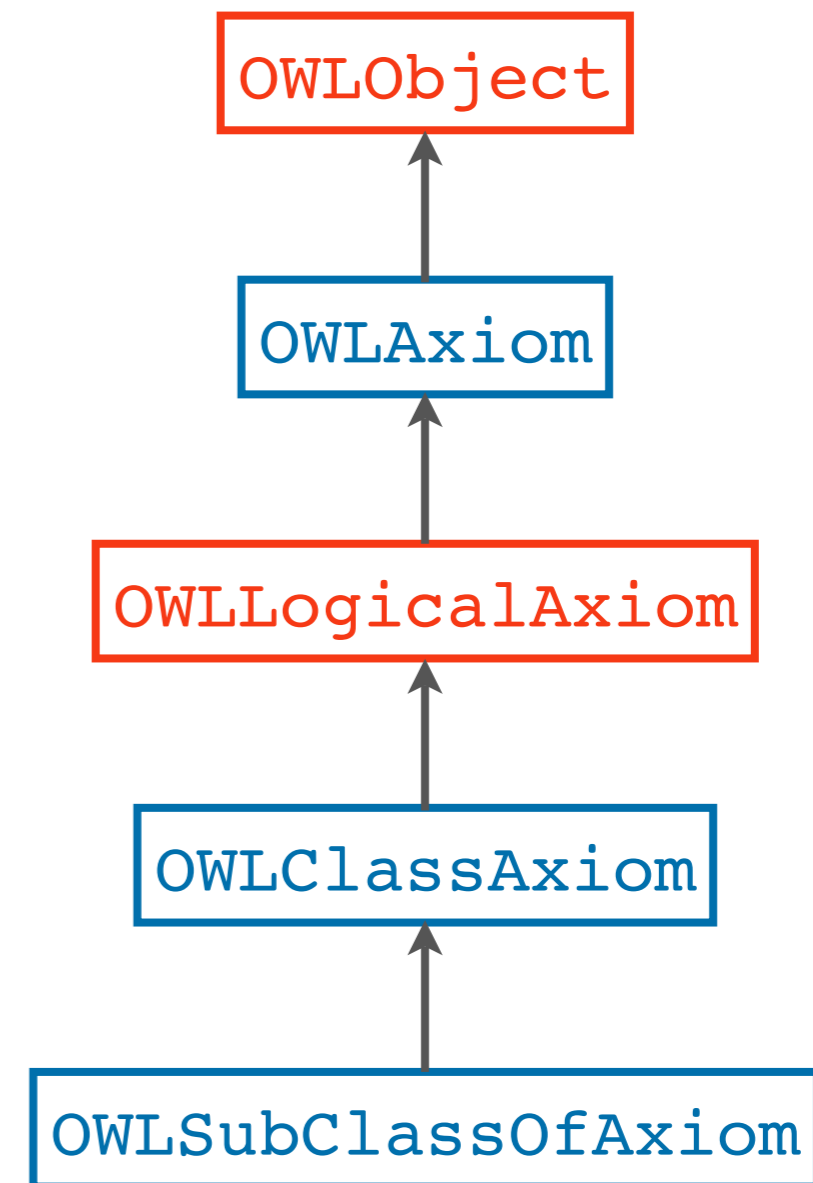
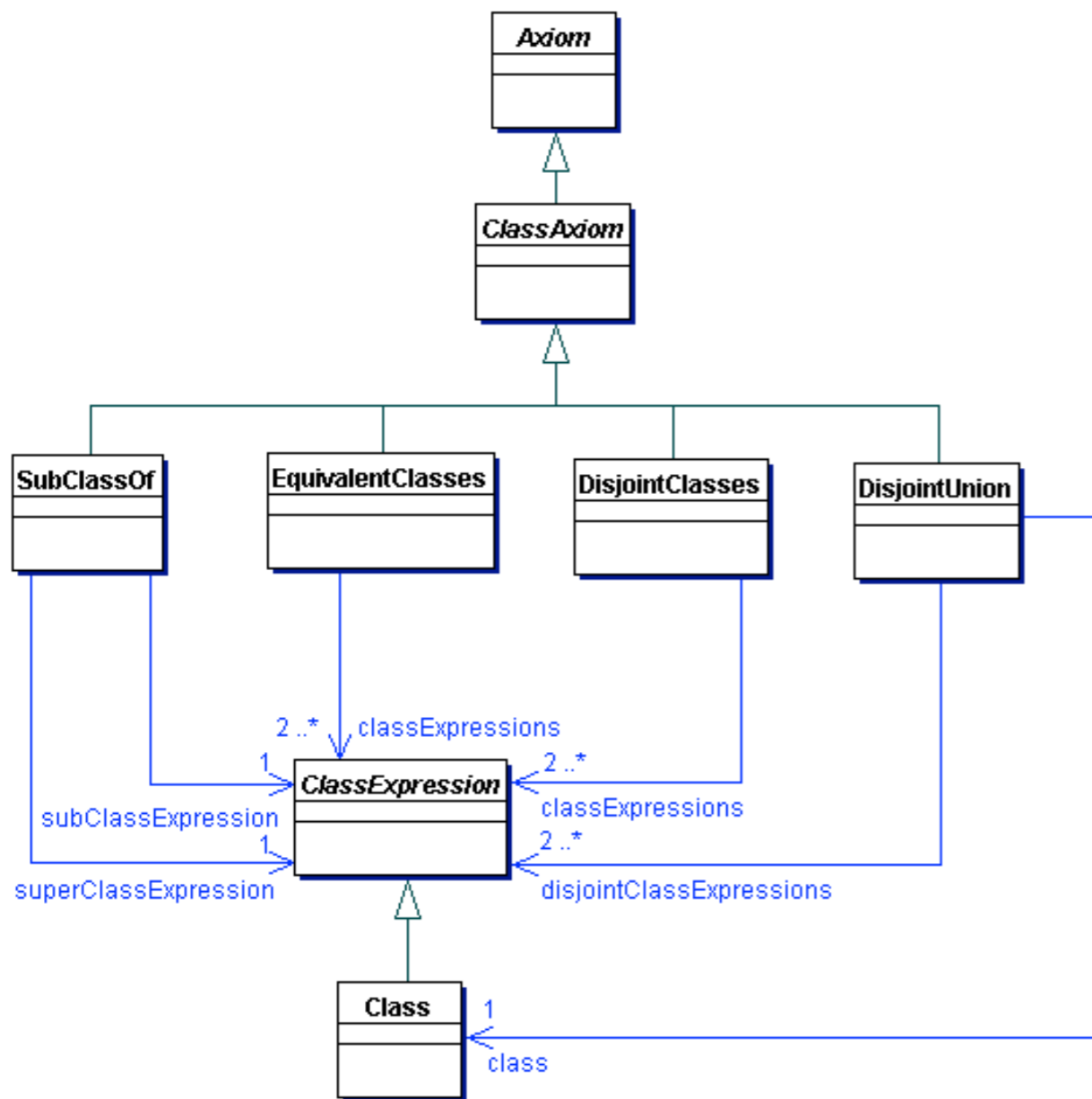
# Design Philosophy



OWL API

Java interfaces  
for manipulating  
OWL 2 ontologies

# From the Structural Specification to Java Interfaces



# Other Machinery

## Ontology management

Creation, loading and saving of ontologies

## Change support

Change objects for axiom addition and removal, listener support

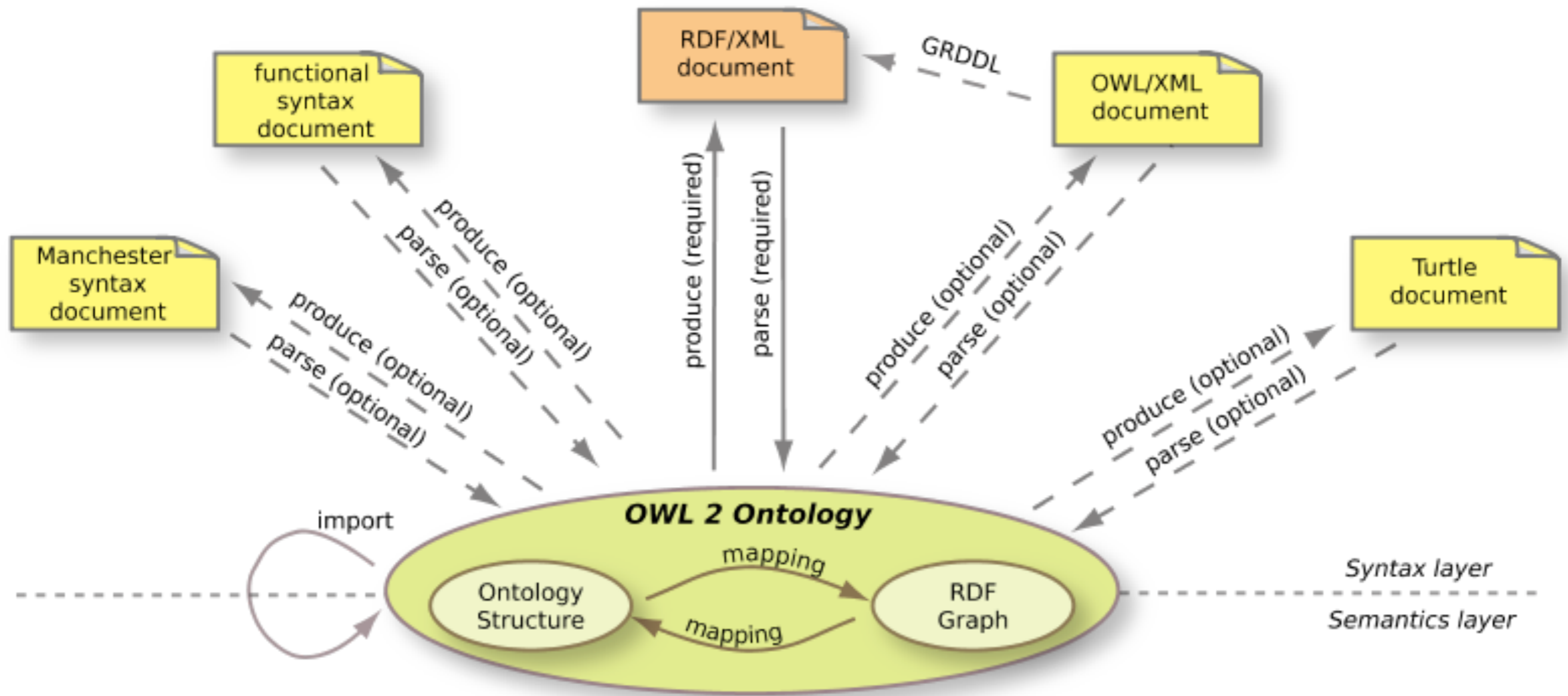
## Common task support

Syntactic validation, metrics, normalisation

## Reasoning support

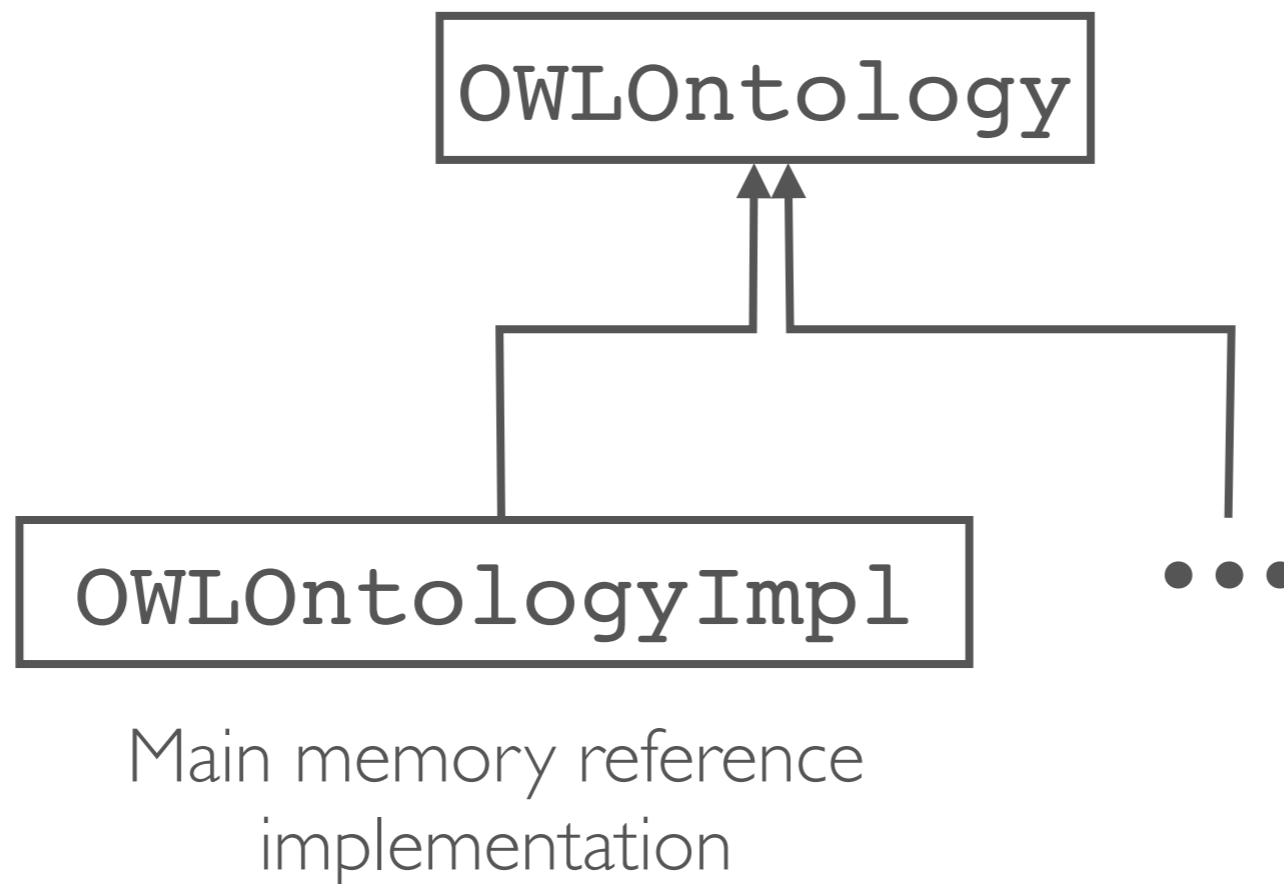
Common reasoner interfaces and reasoner operations

# Concrete Syntaxes



(Courtesy of OWL 2 Web Ontology Language Document Overview)

# Alternative Storage



<http://owldb.sourceforge.net/>



# Reasoning

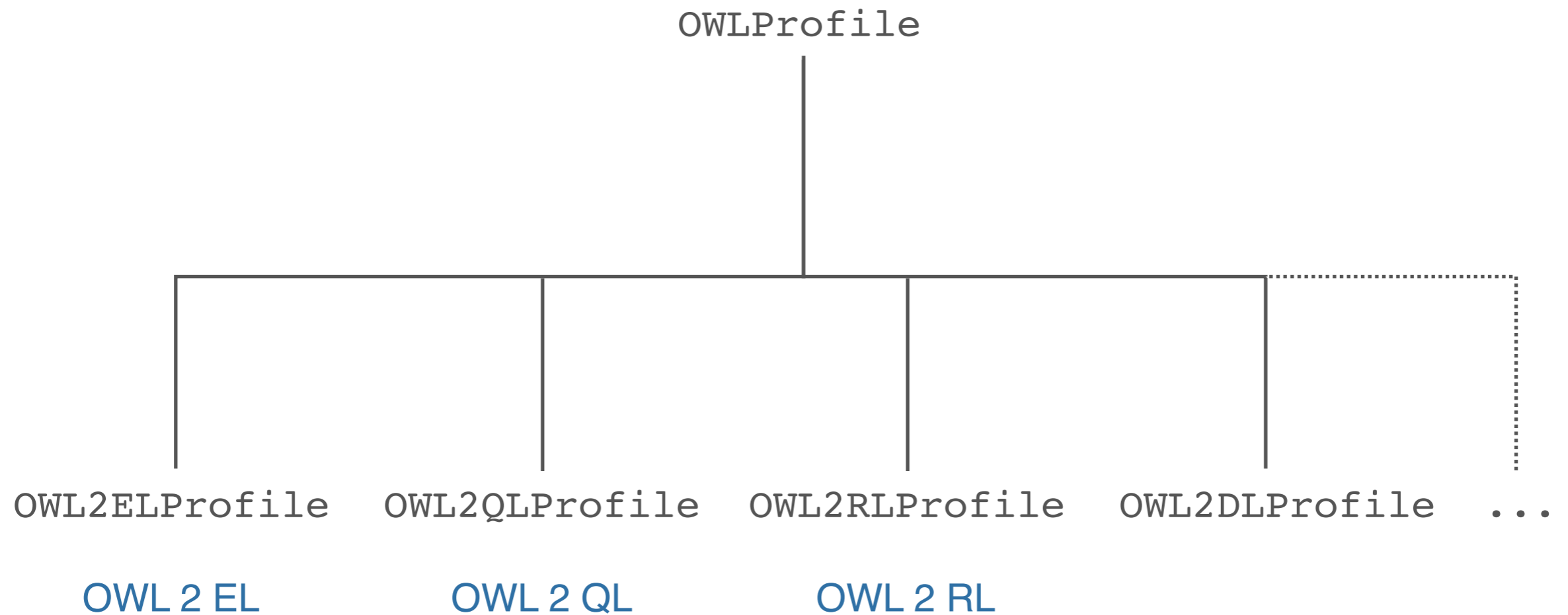
```
public interface OWLReasoner {

    /**
     * Asks the reasoner to interrupt what it is currently doing. An InterruptedException will be thrown in the
     * thread that invoked the last reasoner operation. The OWL API is not thread safe in general, but it is likely
     * that this method will be called from another thread than the event dispatch thread or the thread in which
     * reasoning takes place.
     */
    void interrupt();

    /**
     * A convenience method that determines if the set of reasoner axioms (the set of axioms returned by
     * the {@link #getAxioms()} method) is consistent.
     * @return true if the set of axioms is consistent,
     * or false if the set of axioms is inconsistent.
     * @throws InterruptedException if the reasoning process was interrupted for any particular reason (for example if
     * reasoning was cancelled by a client process)
     */
    boolean isConsistent() throws InterruptedException;

    /**
     * A convenience method that determines if the specified class expression is satisfiable with respect to the
     * set of reasoner axioms (the set of axioms returned by the {@link #getAxioms()} method)
     * @param classExpression The class expression
     * @return true if classExpression is satisfiable with respect to the set of axioms, or
     * false if classExpression is unsatisfiable with respect to the axioms.
     * @throws InconsistentOntologiesException if the reasoner's axiom set is inconsistent
     * @throws EntitiesNotInSignatureException if the signature of the classExpression is not contained within the signature
     * of the reasoner's axiom set.
     * @throws ExpressivenessOutOfScopeException If the class expression contains constructs that are out of the scope
     * of expressiveness that is supported by this reasoner.
     * @throws InterruptedException if the reasoning process was interrupted for any particular reason (for example if
     * reasoning was cancelled by a client process)
     */
    boolean isSatisfiable(OWLClassExpression classExpression) throws InterruptedException;
}
```

# Profiles API



UseOfReservedVocabularyForOntologyIRI

UseOfNonSubClassExpression

UseOfUndeclaredClass



A A

<http://owl.cs.manchester.ac.uk/validator/validate>

Google

# OWL 2 Validation Report

## Summary

**Ontology and imports closure is NOT in OWL 2 EL profile**

## Imports Closure

**Ontology IRI**<http://www.co-ode.org/ontologies/pizza/pizza.owl>**Physical URI**<http://www.co-ode.org/ontologies/pizza/pizza.owl>

## Detailed report

### Use of unsupported class expression

```
SubClassOf(:Rosa ObjectAllValuesFrom(:hasTopping ObjectUnionOf(:GorgonzolaTopping :MozzarellaTopping :TomatoTopping)))
```

### Use of unsupported axiom

```
FunctionalObjectProperty(:hasSpiciness)
```

# Examples of Use

The screenshot shows a web browser window with the following components:

- Address Bar:** `people.owl (file:/Users/seanb/Desktop/Cercedilla2005/hands-on/people.owl)`
- Navigation:** Back, Forward, and Search buttons.
- Active Ontology:** `people.owl (file:/Users/seanb/Desktop/Cercedilla2005/hands-on/people.owl)`
- Tabbed Interface:** `Active Ontology`, `Entities` (selected), and `DL Query`.
- Class Hierarchy:** `Asserted class hierarchy` and `Inferred class hierarchy`. The `elderly` class is selected in the `Asserted class hierarchy` pane.
- Object Property Hierarchy:** `Object property hierarchy` pane showing properties like `drives`, `eaten_by`, `eats`, `has_child`, `has_parent`, `has_part`, `is_pet_of`, `likes`, `part_of`, `reads`, and `works for`.
- Class Annotations:** `Annotations: elderly` pane showing `comment` and `label "elderly"`.
- Description:** `Description: elderly` pane showing `Equivalent classes`, `Superclasses` (including `adult`), `Inferred anonymous superclasses`, `Members` (including `Minnie`), and `Disjoint classes`.

Ontology metrics: ☰ ☱ ☲ ☳

**Metrics**

Class count	61
Object property count	14
Data property count	0
Individual count	22
DL expressivity	ALCHOIN

**Class axioms**

SubClass axioms count	34
Equivalent classes axioms count	22
Disjoint classes axioms count	4
GCI count	1
Hidden GCI Count	4

**Object property axioms**

Sub object property axioms count	3
Equivalent object properties axioms co...	0
Inverse object properties axioms count	3
Disjoint object properties axioms count	0
Functional object property axioms count	0
Inverse functional object property axio...	0
Transitive object property axioms count	0
Symmetric object property axioms count	0
Anti-symmetric object property axioms	0

# Metrics API for ontology metrics view

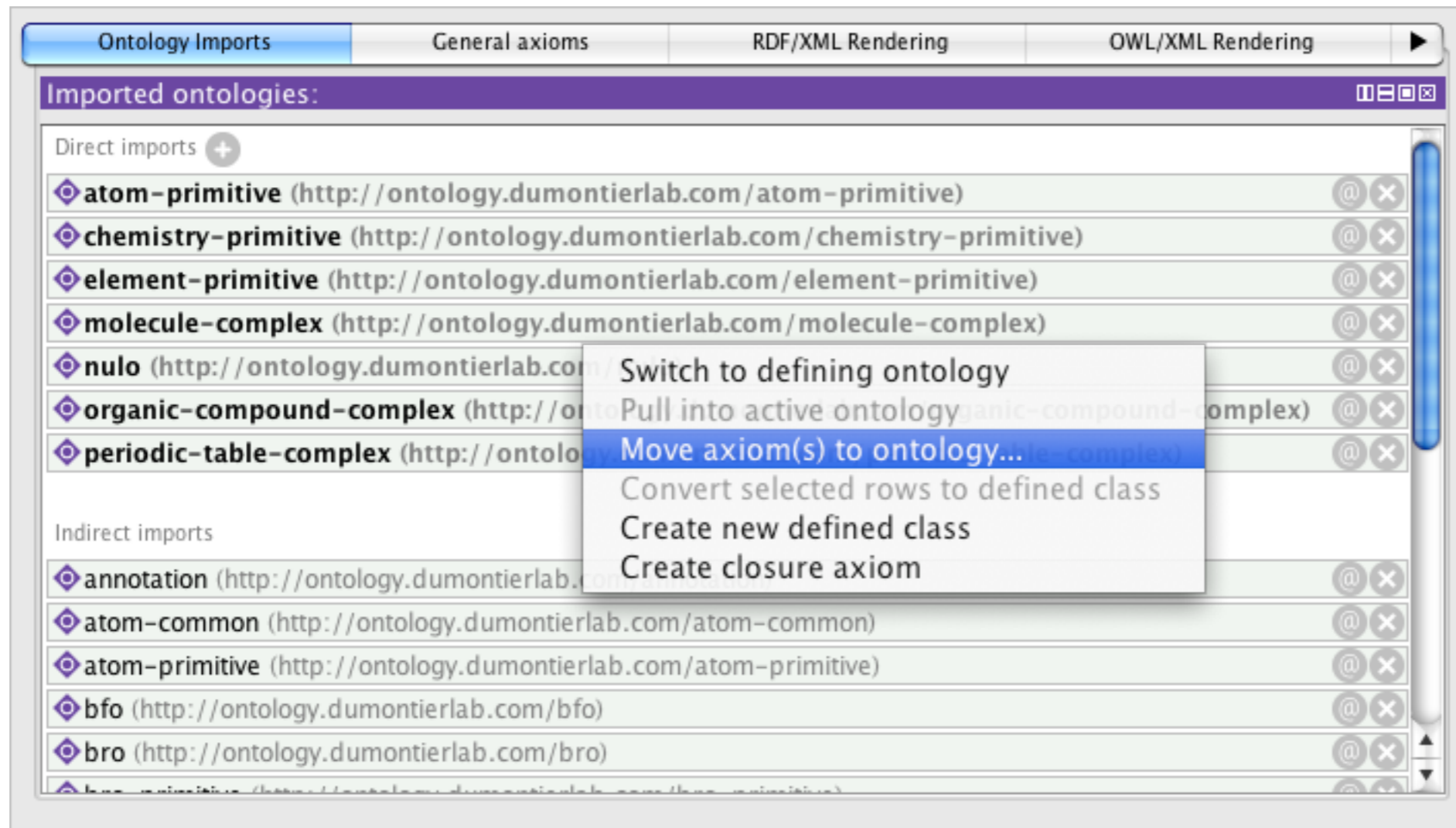
Class Annotations Class Usage

Usage: animal

Show:  this  disjoints  named sub/superclasses

- pet\_owner **equivalentTo** person  
and has\_pet **some** animal
- ▼ ● sheep
  - sheep **subClassOf** animal
- ▼ ● tiger
  - tiger **subClassOf** animal
- ▼ ● vegetarian
  - vegetarian **equivalentTo** animal  
and eats **only** (not animal)  
and eats **only** (not (part\_of **some** animal))
- ▼ ■ eats
  - eats **domain** animal

Axiom retrieval by signature  
and type for usage views

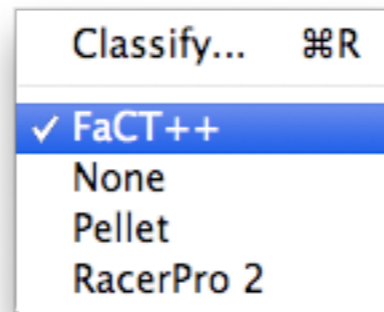


Working with multiple ontologies  
in an imports closure

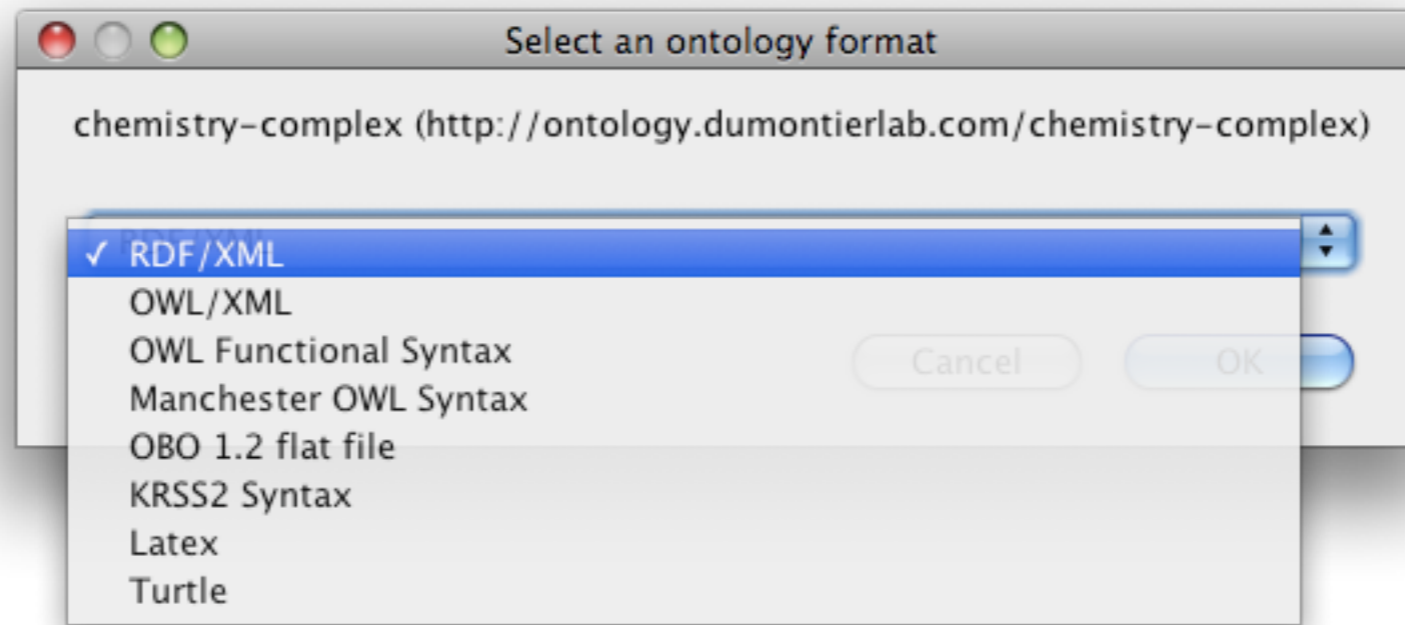
Undo	⌘Z
Redo	⇧⌘Z
Cut	⌘X
Copy	⌘C
Paste	⌘V
Delete ...	⌘⌫
Find in view...	⌘F
Create new	⌘N
Create child	⌘\
Create sibling	⌘/
Duplicate selected class...	⇧⌘C
Convert to primitive class	⌘P
Convert to defined class	⌘D
Add covering axiom	
Make all individuals distinct...	
Make primitive siblings disjoint	⌘J
Remove disjoints for subclasses...	
Remove all disjoint axioms...	
Prefixes...	⇧⌘P

Change objects used for undo/redo

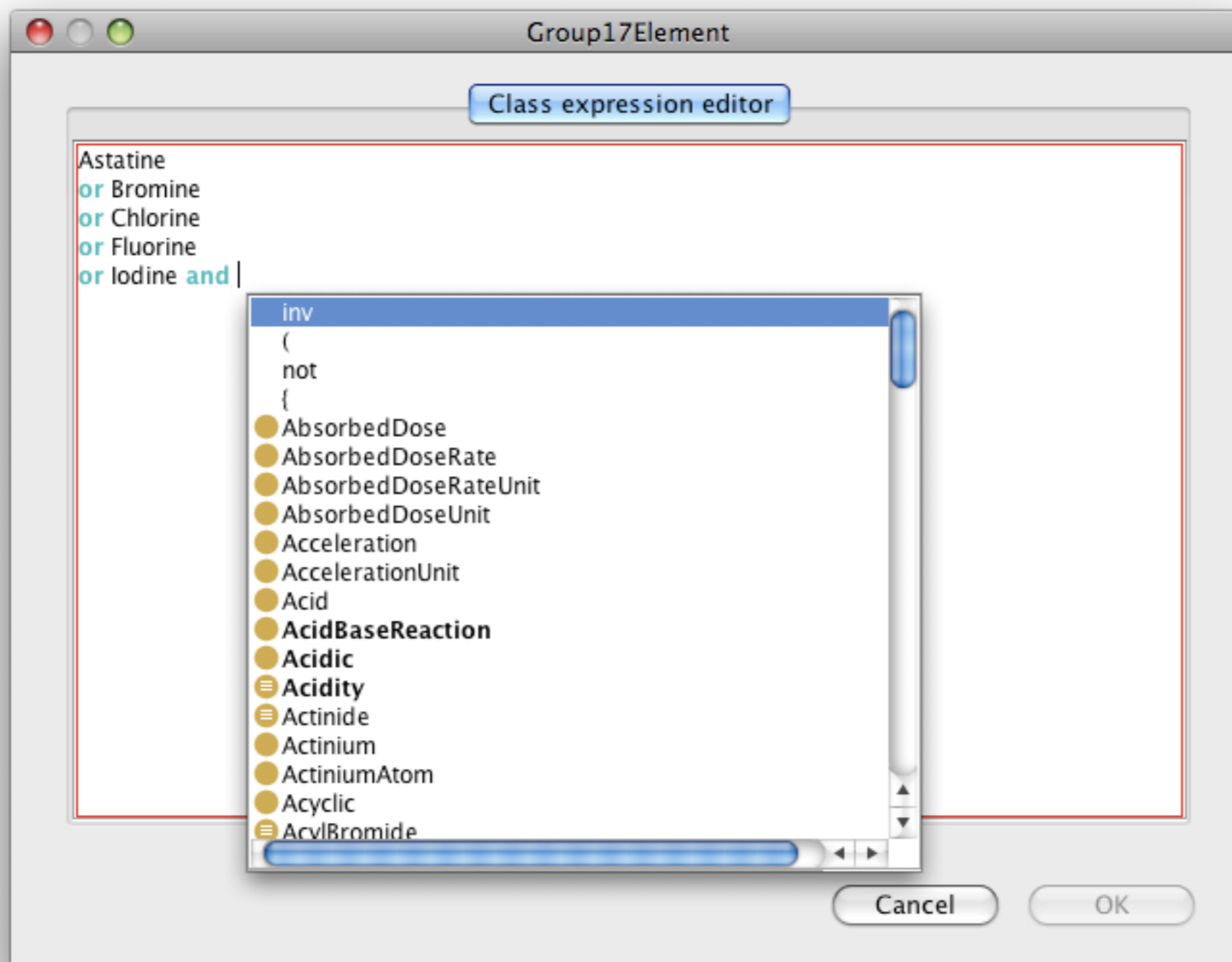




Common reasoner interface



Out of the box serialisation support  
for multiple file formats



Use of Manchester Syntax parser to provide editor autocompletion and syntax checking

Explanation workbench

Show regular justifications       All justifications  
 Show laconic justifications       Limit justifications to

Entailment	No. Justi...
<b>mad_cow</b>	1
<b>Nothing</b>	-
animal_lover SubClassOf pet_owner	1
cat_liker SubClassOf person	1
cat_owner SubClassOf pet_owner	2

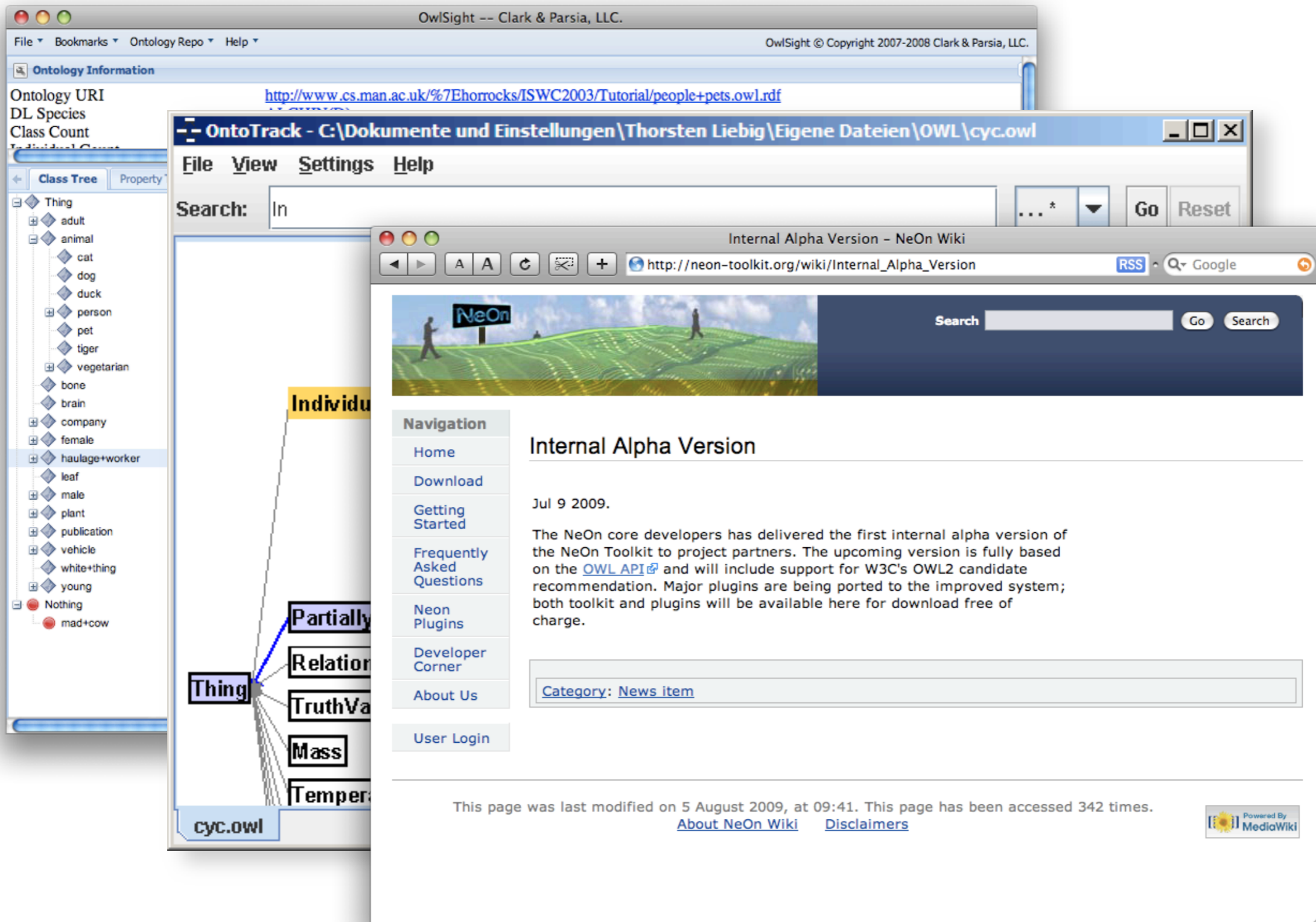
Explanation 1 (Entailment 1)  Display laconic explanation

	cat_owner SubClassOf pet_owner		
1)	cat_owner EquivalentTo person and (has_pet some cat)	2	<input type="checkbox"/>
2)	cat SubClassOf animal	1	<input type="checkbox"/>
3)	pet_owner EquivalentTo person and (has_pet some animal)	2	<input type="checkbox"/>

Explanation 2 (Entailment 1)  Display laconic explanation

	cat_owner SubClassOf pet_owner		
1)	cat_owner EquivalentTo person and (has_pet some cat)	2	<input type="checkbox"/>
2)	has_pet Range animal	1	<input type="checkbox"/>
3)	pet_owner EquivalentTo person and (has_pet some animal)	2	<input type="checkbox"/>

Axiom oriented model simplifies many tasks



Widely used in many other tools

# Acknowledgements



Sean Bechhofer

# Acknowledgements



Birte Glimm



Boris Motik



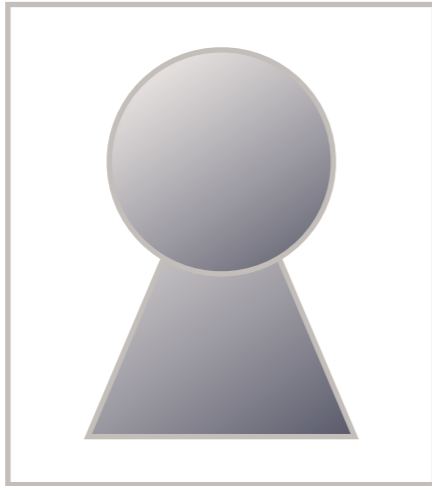
Tim Redmond



Ignazio Palmisano



# Acknowledgements



Ron Alford



Nick Drummond



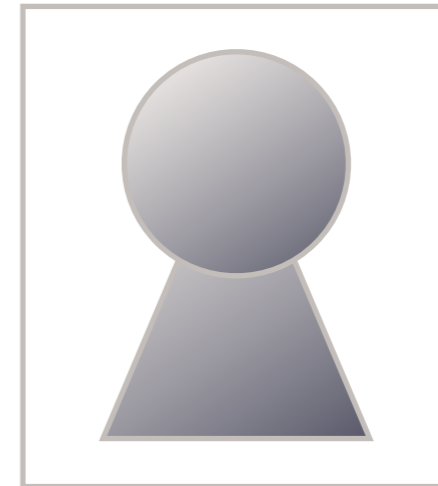
Olaf Noppens



Thomas Schneider



Evren Sirin



Mike Smith