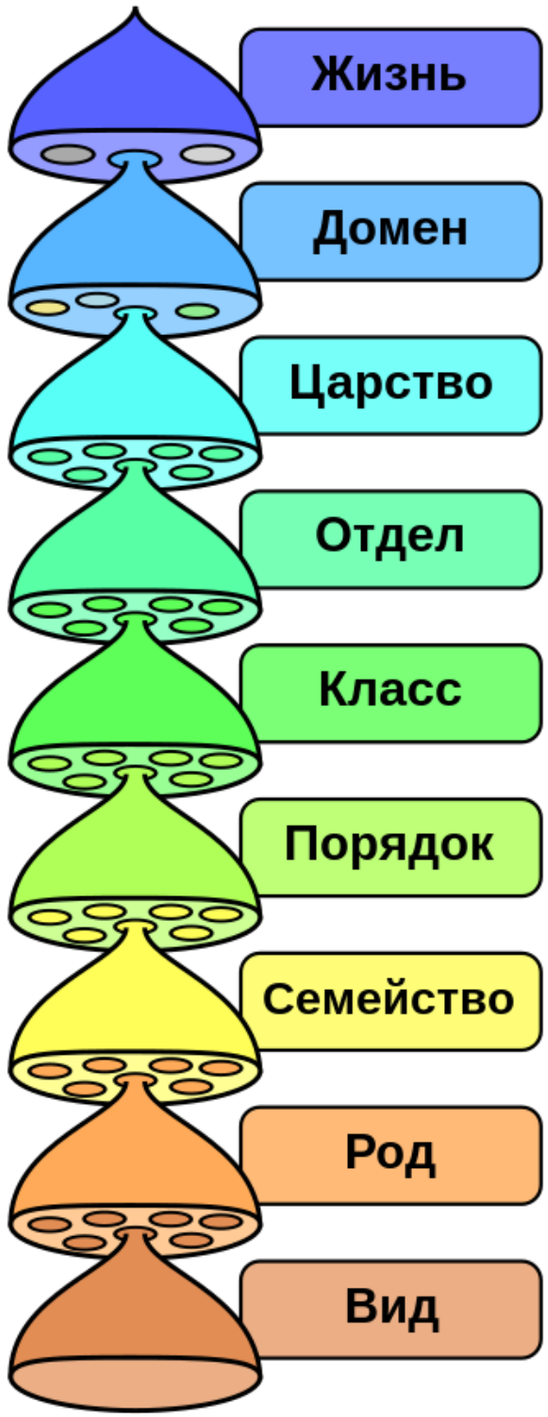
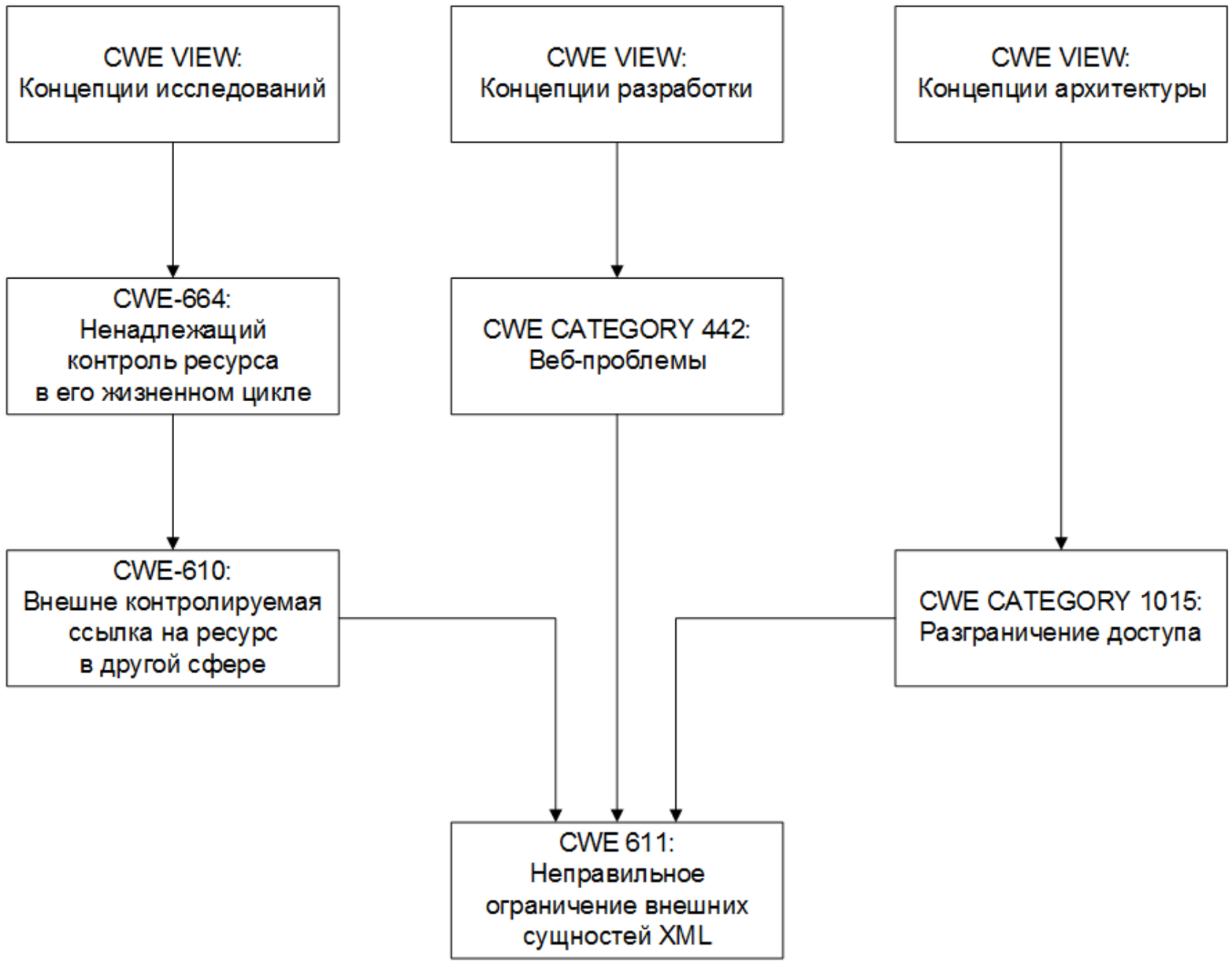
Онтологическое моделирование

Практические соображения

Онтологии среди нас

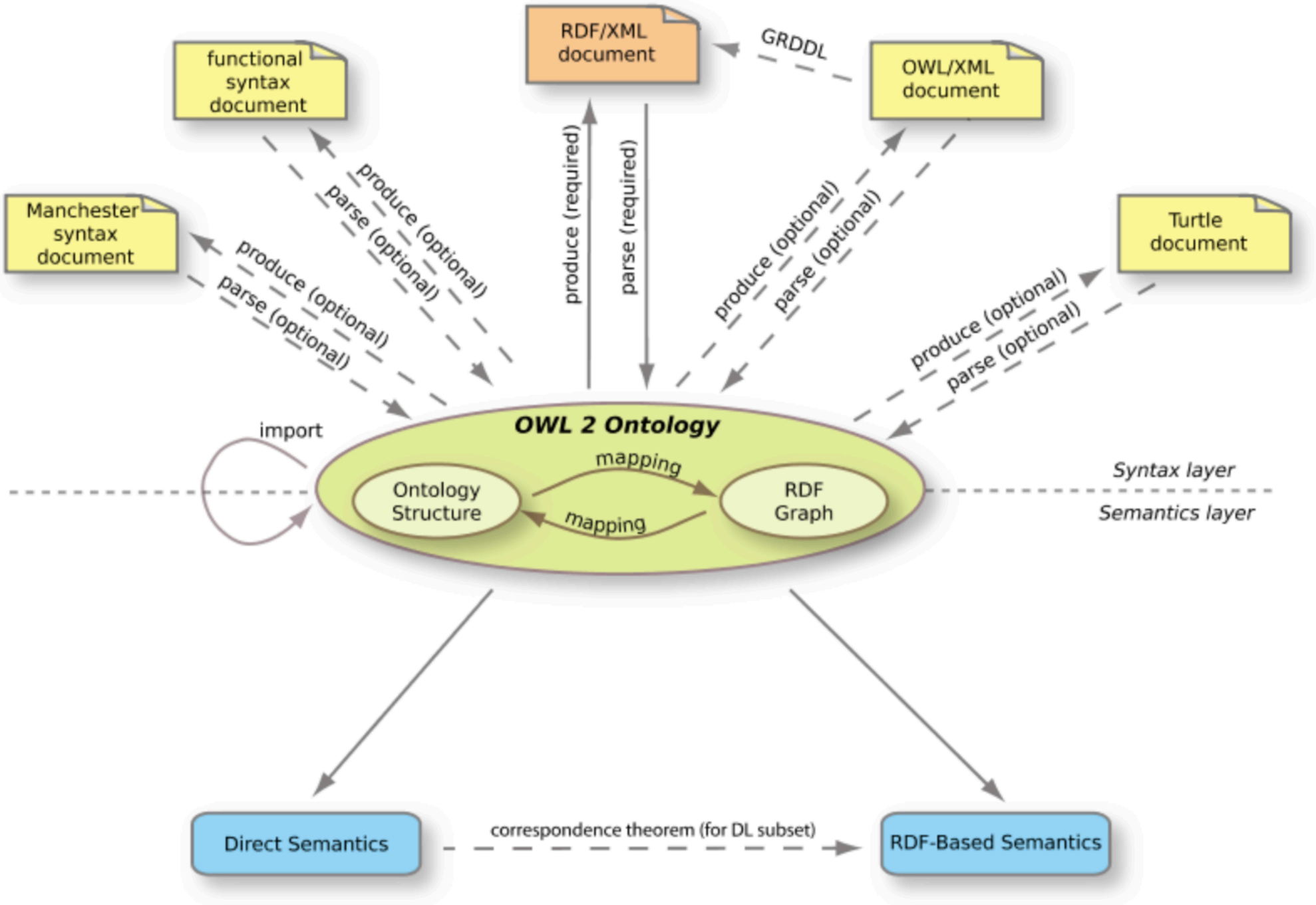




Язык OWL и диалекты

OWL - язык описания онтологий.

OWL 2 Full - Максимальная выразительность
OWL 2 DL - Для быстрых вычислений и выразительности
OWL 2 EL - Для быстрых вычислений с большим количеством классов
OWL 2 QL - Для быстрых вычислений с большим количеством индивидов
OWL 2 RL - Для языков правил


RDF/XML document
GRDDL
OWL/XML document
functional syntax document
Manchester syntax document
Turtle document
produce (required)
parse (required)
produce (optional)
parse (optional)
import
OWL 2 Ontology
mapping
Ontology Structure
RDF Graph
Syntax layer
Semantics layer
Direct Semantics
correspondence theorem (for DL subset)
RDF-Based Semantics

Из чего состоит онтология?

1. **Классы - объекты предметной области**
2. ***(Атрибуты классов)***
3. **Свойства (отношения) - взаимосвязи между понятиями**
4. ***(Ограничения свойств)***

Этапы проектирования онтологии

«Проектируйте онтологии так, как хотите» - Дж. Стетхем

1. **Определение области и масштаба онтологии**
2. **Формулирование вопросов для проверки компетенции**
3. **Поиск существующих решений**
4. **Перечисление важных терминов**
5. **Определение классов и их иерархии**
6. **Определение свойств и их ограничений**
7. **Создание экземпляров (индивидов)**

Иерархия классов

1. **Наследование это is_a, не kind_of**
2. **Наследование транзитивно**
3. **Циклы классов = эквивалентность классов**
4. **Узлы братья представляют один уровень общения**
5. **Новое свойство иногда лучше нового класса**
6. **Экземпляр отличается от класса только степенью детализации**

Свойства

1. **У каждого свойства определены Domain и Range**
2. **Значения некоторых свойств - литералы (DataProperties)**
3. **Свойства бывают транзитивные**
4. **Свойства бывают функциональные**
5. **Для свойств можно определить ограничения мощности**

Где логика?

		DL syntax	Protégé	Python + Owlready2	First-order logic	Semantics in set formula
Const.	Top	$\top$	Thing	Thing	$\top$, such as $\forall x, \top(x) = true$	Δ
	Bottom	$\bot$	Nothing	Nothing	$\bot$, such as $\forall x, \bot(x) = false$	$\emptyset$
Axioms	Subsumption	$A \sqsubseteq B$	A subclass of B	class A(B): ... (assertion) A.is_a.append(B) (assertion) issubclass(A, B) (test)	$\forall x, A(x) \rightarrow B(x)$	$f(A) \subseteq f(B)$
		$R \sqsubseteq S$	R subproperty of S	(same as above)	$\forall x \forall y, R(x,y) \rightarrow S(x,y)$	$f(R) \subseteq f(S)$
	Equivalence	$A \equiv B$	A equivalent to B	A.equivalent_to.append(B) (as.) B in A.equivalent_to (test)	$\forall x, A(x) \leftrightarrow B(x)$	$f(A) = f(B)$
	Instanciation	$A(i)$	i type A	i = A() (assertion) i.is_instance_of.append(A) isinstance(i, A) (test)	$A(i)$	$f(i) \in f(A)$
	Relations	$R(i,j)$	i object property assertion j i data property assertion j	i.R = j (R is functional) i.R.append(j) (otherwise)	$R(i,j)$	$(f(i), f(j)) \in f(R)$
Semantic connectors	Complement	$\neg A$	not A	Not(A)	$\neg A(x)$	$\Delta \backslash f(A)$
	Intersection	$A \sqcap B$	A and B	A & B (or) And([A, B,...])	$A(x) \wedge B(x)$	$f(A) \cap f(B)$
	Union	$A \sqcup B$	A or B	A | B (or) Or([A, B,...])	$A(x) \vee B(x)$	$f(A) \cup f(B)$
	Extension	$i, j, ...$	{i, j,...}	OneOf([i, j,...])	$x \in \{i, j, ...\}$	$\{f(i), f(j), ...\}$
	Inverse	R^-	inverse of R	Inverse(R) (construct) S.inverse = R (assertion)	$\forall i \forall j, S(i,j) = R(j,i)$	$\{(a,b) \mid (b,a) \in f(R)\}$
	Transitive closure	R^+	-	-		$\cup_{i \geq 1}(f(R))^i$
	Composition	$R \circ S$	R o S	PropertyChain([R, S])		$\{(a,c) \in \Delta \times \Delta \mid \exists b, (a,b) \in f(R) \wedge (b,c) \in f(S$
	Existential quantifier	$\exists R.B$	R some B	R.some(B)	$\exists y, R(x,y) \wedge B(y)$	$\{a \in \Delta \mid \exists b, (a,b) \in f(R) \wedge b \in f(B)\}$
	Universal quantifier	$\forall R.B$	R only B	R.only(B)	$\forall y, R(x,y) \rightarrow B(y)$	$\{a \in \Delta \mid \forall b, (a,b) \in f(R) \rightarrow b \in f(B)\}$
	Number restrictions	$= 2R.B$	R exactly 2 B	R.exactly(2, B)	$\lvert\{y \mid R(x,y) \wedge B(y)\}\rvert = 2$	$\{a \in \Delta \mid \lvert\{b \mid (a,b) \in f(R) \wedge b \in f(B)\}\rvert = 2\}$
		$\leq 2R.B$	R max 2 B	R.max(2, B)	$\lvert\{y \mid R(x,y) \wedge B(y)\}\rvert \leq 2$	$\{a \in \Delta \mid \lvert\{b \mid (a,b) \in f(R) \wedge b \in f(B)\}\rvert \leq 2\}$
		$\geq 2R.B$	R min 2 B	R.min(2, B)	$\lvert\{y \mid R(x,y) \wedge B(y)\}\rvert \geq 2$	$\{a \in \Delta \mid \lvert\{b \mid (a,b) \in f(R) \wedge b \in f(B)\}\rvert \geq 2\}$
	Role filler	$\exists R.\{j\}$	R value j	R.value(j)	$R(x,j)$	$\{a \in \Delta \mid (a, f(j)) \in f(R)\}$
Decomposable	Disjoint	$A \sqcap B \sqsubseteq \bot$	A disjoint with B	AllDisjoint([A, B])	$\forall x, \neg(A(x) \wedge B(x))$	$f(A) \cap f(B) = \emptyset$
	Property domain	$\exists R. \top \sqsubseteq A$	R domain A	R.domain = [A]	$\forall x, (\exists y, R(x,y)) \rightarrow A(x)$	$f(R) \subseteq \{(a,b) \mid a \in f(A)\}$
	Property range	$\top \sqsubseteq \forall R.B$	R range B	R.range = [B]	$\forall x \forall y, R(x,y) \rightarrow B(y)$	$f(R) \subseteq \{(a,b) \mid b \in f(B)\}$
	Role filler as class property	$A \sqsubseteq \exists R.\{j\} \sqcap (\exists R^-.A)(j)$	-	A.R = j (R is functional) A.R.append(j) (otherwise)	-	-
	Local closed world	-	-	close_world(A)	-	-

Owlready2

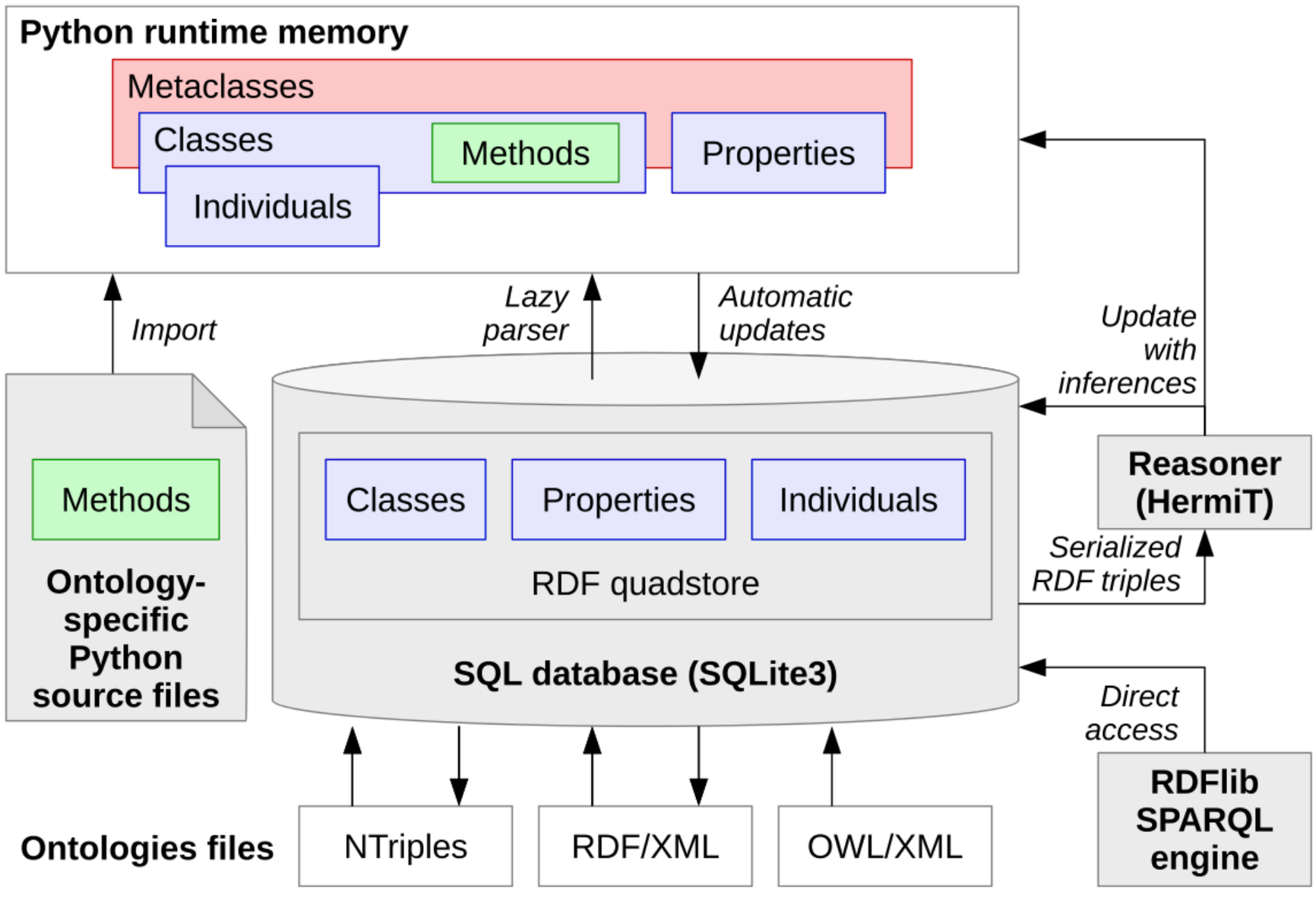
Онтологически-ориентированное программирование

Общая характеристика

- **Owlready2 - пакет для онтологического моделирования на python 3.**
- **Позволяет манипулировать существующими и создавать новые онтологии, а также запускать логический вывод (HermeT или Pellet)**
- **Мотивация разработчиков:**
  - **Разработка простого программного интерфейса для онтологий**
  - **Реализация возможности манипулирования классами**
  - **Реализация вывода для модели локального закрытого мира**

Lamy JB. Owlready: Ontology-oriented programming in Python with automatic classification and high level constructs for biomedical ontologies.
Artificial Intelligence In Medicine 2017;80:11-28

Архитектура owlready



Программные интерфейсы для онтологий

Интерфейс	Недостатки
SPARQL	Требуется новый запрос для каждого доступа RDF не OWL (Почти) Отсутсвует возможность манипулирования классами
OWLAPI/Jena	Сложность Объект онтологии - экземпляр, а не класс
OOP: GO!, модули для JAVA, C#, Lisp	Для статической объектной модели не возможен вывод Незрелость инструментов

Соответствие между объектной и онтологической моделями

Онтологическая модель	Объектная модель owlready
Онтология	Модуль
Класс	Подкласс мета-класса Thing
Индивид	Экземпляр
Роль	Подкласс мета-класса Property
Литералы	Соответсвующие встроенные типы

Открытый и закрытый мир

Истинность не зависит от наших знаний.

Известно: Я люблю пиццу. Люблю ли я рассказывать про owlready?
Открытый мир: Неизвестно
Закрытый мир: Нет

Resonerы работают с предположением открытого мира.

Известно: Пациент диабетик.
Про лекарство не известно, что оно противопоказанно при диабете.
Лекарство можно прописывать, если пациент не имеет противопоказаний к нему.
Прописывать лекарство?

Открытый мир: Нет
Закрытый мир: Да

Локальный закрытый мир в owlready

Проблему можно решить дополнительными утверждениями:

Известно: Пациент диабетик.
Про лекарство не известно, что оно противопоказанно при диабете.
Про лекарство известно, что оно имеет ровно n указанных противопоказаний.
Лекарство можно прописывать, если пациент не имеет противопоказаний к нему.
Прописывать лекарство?

Открытый мир: Да
Закрытый мир: Да

Функция close_world(Class| Individual):

Для индивида: все отношения и все их значения известны

Для класса: все индивиды и подклассы известны, Все индивиды и подклассы также рассматриваются в закрытом мире

План

1. **Язык OWL и его диалекты**
2. **Объекты OWL**
3. **Математическая логика и OWL**
4. **Примеры практического использования онтологий**
5. **Этапы создания онтологии**
6. **Онтология вин**
7. **Онтология SNOMED**
8. **Библиотека Owlready2 и ее использование**