Фреймворки обработки онтологий

Сюда же включаю все парсеры и инструменты сериализации.
Хочется: фреймворк по типу Jena с полным представлением онтологии в виде объектов

C++

Пока ничего не нашел

C

- Raptor RDF Parser Library - набор парсеров и сериалайзеров, поддержавающий RDF-триплеты и расширения (из известных мне - Dublin Core, OWL). Представление - какие-то гнусные "объекты Си" - порождение извращенных фантазий какого-то фиолетового осьминога.

Утверждают, что есть отличная интеграция с Rasqal RDF Query Library(запросы к RDF) и Redland Language Bindings (API на разных языках, в т.ч. Python). Синтаксис - своеобразный.

```
#include <stdio.h>
#include <raptor.h>

/* rdfprint.c: print triples from parsing RDF/XML */

void
print_triple(void* user_data, const raptor_statement* triple)
{
  raptor_print_statement_as_ntriples(triple, stdout);
  fputc('\n', stdout);
}

int
main(int argc, char *argv[])
{
  raptor_parser* rdf_parser=NULL;
  unsigned char *uri_string;
  raptor_uri *uri, *base_uri;

  raptor_init();

  rdf_parser=raptor_new_parser("rdfxml");

  raptor_set_statement_handler(rdf_parser, NULL, print_triple);

  uri_string=raptor_uri_filename_to_uri_string(argv[1]);
  uri=raptor_new_uri(uri_string);
```

```
  base_uri=raptor_uri_copy(uri);

  raptor_parse_file(rdf_parser, uri, base_uri);

  raptor_free_parser(rdf_parser);

  raptor_free_uri(base_uri);
  raptor_free_uri(uri);
  raptor_free_memory(uri_string);

  raptor_finish();
}
```

Python

- Бритва Эйлера - поддерживающий нотацию N3 движок вывода.

Пока не понял, но похоже на нужное

```
[[http://rdflib.net/]]
```

OWL Logic - Что-то про Python OWL API и кучу других полезных OWL-штучек. Вот тут - исходники на ру