## Разработка методов и средств на базе онтологического представления экспертных знаний для поддержки процесса решения задач компьютерного моделирования

Главная идея работы – организовать все знания в виде онтологий, а именно в виде owl+xml описаний. Далее по этим онтологиям можно делать различные запросы на специальном языке SPARQL (который немного напоминает SQL для баз данных). Сейчас существует несколько программных решений, которые позволяют парсить такие описания онтологий, а также исполнять запросы на SPARQL. Одним из таких решений является программа с открытым кодом pellet, написанная на Java. Сейчас pellet позволяет работать только через командную строку, таким образом на каждый запрос к онтологии производится репарсинг всей онтологии из owl файла, что занимает примерно 30 секунд.

Основными идеями в работе является взять за основу уже готовый парсер из pellet, в качестве frontend-а использовать web-сервис со стандартными транспортными протоколами SOAP+WSDL. Таким образом доступ к знаниям можно осуществлять из других языков (основная часть системы написана на C#).

Кроме этого, основной особенностью будет является организация сессионной работы с этим сервисом, и самое главное, возможность дополнять базовую онтологию пользовательскими знаниями. Идея такая: пользователь системы, который зашел в систему с целью расчета каких-то квантово-химических и физических характеристик своей атомно-молекулярной структуры, с помощью вопросов системы взаимодействия с пользователем генерирует свое представление (иногда очень смутное) о том, что ему нужно рассчитать в виде онтологий. Далее он свое представление в виде owl передает в этот сервис, там берется уже загруженная базовая онтология, она дополняется пользовательским расширением. Далее, система делает SPARQL запросы к получившейся расширенной пользовательской онтологии, с помощью которых понимает, что именно и как вычислять.

Сейчас уже реализован web-сервис, который запущен под application server Glassfish 3.0.1. В web-сервисе реализованы методы для создания и закрытия сессий, исполнения запросов к онтологии, которая храниться в пользовательской сессии на сервере. Также реализован метод для дополнения базовой онтологии. Расширенная онтология сохраняется в сессии, таким образом ее можно дополнять несколькими вызовами метода web-сервиса.

Диаграмма основных классов, используемых в PelletService представлена на рис. 1.

Фронтендом в приложении является класс PelletService, в котором с использованием Java аннотаций описаны интерфейс web-сервиса и web-методы:

|  |
| --- |
| @WebService()  interface {  String createSession();  String executeQuery(String sessionId, String sparql);  int addOWLModel(String sessionId, String content);  int closeSession(String sessionId);  } |

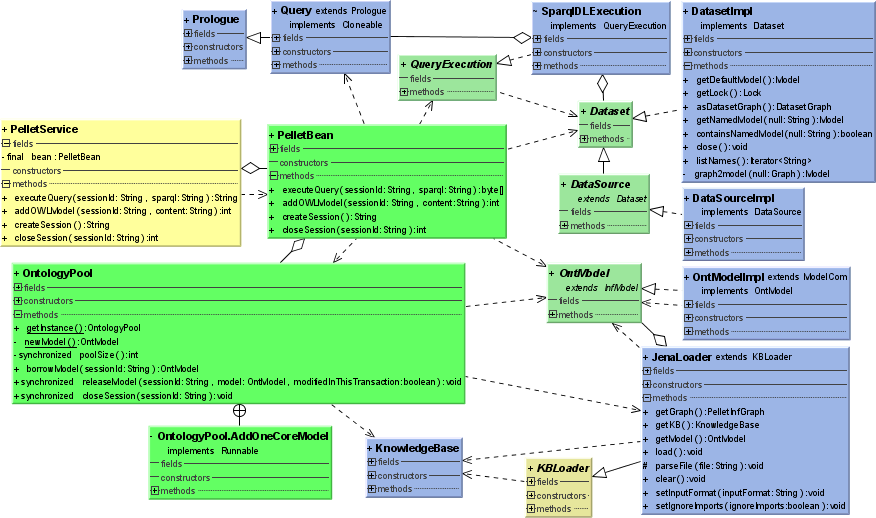


Рис 1. Диаграмма классов

## Как запустить приложение

Приложение распространяется, как архив pellet.war с web-сервисом и всеми необходимыми библиотеками (pellet, jena, etc.) внутри. Этот web-архив можно запустить под любым application сервером, в котором существует функциональность по запуску web-сервисов.

Далее по пунктам будет описано, как запустить приложение под бесплатно-распространяемым application сервером glassfish, который разрабатывается компанией Oracle (ранее в Sun).

1. Скачать glassfish 3.0.1 application server с сайта Oracle.
2. Установить glassfish.
3. В папке bin, куда будет установлен glassfish, запустить startserv.bat.
4. Запуск может занять минуты 2-3, после надписи Welcome to FELIX, должны появиться еще записи в логи.
5. После этого в любом браузере перейти на localhost:8080, там перейти по линку go to administration page.
6. Administration page грузится первый раз очень долго (иногда может понадобиться в папке bin запустить stopserv.bat, после этого повторить пункты 3-5 заново).
7. На administration page нажать на виджет Deploy web application, указать путь до pellet.war и нажать OK. Далее PelletService должен появиться в списке приложений.
8. В качестве полезных инструментов, glassfish позволяет запустить Tester для задеплойеного web-сервиса, в котором можно с web-страницы повызывать методы web-сервиса.
9. В качестве проверки, что приложение успешно работает внутри application сервера, можно выполнить какой-нибудь тестовый запрос к базовой онтологии, через метод executeQuery.