

Н.С. Константинова

**КРИТЕРИИ ВЫБОРА СРЕДСТВ
ДЛЯ ПРЕДСТАВЛЕНИЯ ОНТОЛОГИЙ
(на материале ПО «Финансовый менеджмент»)**

1. Введение

Данное исследование направлено на построение онтологии для предметной области (ПО) «Финансовый менеджмент», при этом особое внимание было уделено подобласти «Финансовые риски». В ходе работы над созданием данной онтологии использовались несколько онторедакторов, таким образом, была возможность сравнить их функциональность и удобство для решения прикладных задач.

Онторедакторы представляют собой специальные программы, обеспечивающие удобный интерфейс для разработчиков онтологий. Их главная задача – облегчить и ускорить процесс создания онтологий. Однако для наиболее эффективного использования онторедактора необходимо изначально определиться со спецификой обрабатываемой области и в соответствии с этим подобрать программу, подходящую для создаваемой онтологии. Во многом оценка функциональности программ для построения онтологий зависит от следующих факторов:

- практическая задача (цели разработчика);
- область знания, в рамках которой строится онтология;
- разрабатываемая онтология¹.

Сейчас существует множество коммерческих и свободно распространяемых программных продуктов для создания и редактирования онтологий. Целый ряд инструментов позволяет также

¹ *Добров Б.В., Иванов В.В., Лукашевич Н.В., Соловьев В.Д.*
Онтологии и тезаурусы: Учебный курс // URL: <http://download.yandex.ru/class/solovyev/plan.pdf>

поддерживать документирование онтологий, импорт и экспорт в различные форматы, а также выполнять другие функции. В ходе исследования велась работа с двумя онторедакторами: DOE и Protégé. В статье обсуждаются особенности данных средств в отношении их пригодности для достижения поставленной цели.

2. Основные черты онторедкторов DOE и Protégé

2.1. DOE

Прежде всего, следует обратить внимание на специфику внутренней организации DOE, где присутствуют:

- классы;
- свойства;
- домены.

В основе инструмента DOE лежит методология построения онтологий, разработанная Р. Тронси и А. Исааком¹. Данные исследователи отказались от использования существующих методов в связи с тем, что они не давали возможность строго и точно структурировать онтологии, в частности, подробно описывать родовидовые отношения. Использование предложенной авторами методологии должно обязать «создателя онтологий выразить значение, которое часто специфицируется только в качестве дополнительного комментария»². Их предложение заключалось в прохождении нескольких этапов при конструировании онтологии. На первой стадии предлагается подбирать понятия, предварительно устанавливая место вводимого понятия в родовидовой иерархии. Далее выделяются четыре дифференциальных принципа, описывающие взаимоотношения между понятиями.

¹ *Troncy R., Isaac A.* DOE: une mise en oeuvre d'une méthode de structuration différentielle pour les ontologies // 13èmes Journées Franco-phones d'Ingénierie des Connaissances: IC'2002. 28–30 May 2002, Rouen, France // URL: <http://opales.ina.fr/public/papers/PresentationIC02.pdf>

² Там же.

Исследователи призывают формулировать сходства и различия с родовым понятием (отцом), а также сходства и различия с координируемыми понятиями (братьями). В результате этих процедур получается подробное описание значения понятия и его определение. Наряду с построением деревьев понятий на данном этапе формируются и деревья отношений. В рамках этого шага предполагается также добавление дополнительной лингвоспецифичной информации (для конкретного языка: предпочтительный термин, энциклопедическое определение, синонимы). Результатом описанных действий является создание так называемой дифференциальной онтологии (откуда и название редактора – DOE, Differential Ontology Editor). На следующем этапе разработчики, пользующиеся DOE, получают возможность сформировать еще одну, так называемую референциальную онтологию, в которую включаются отдельные экземпляры, дополняются и конкретизируются отношения. Описанной методологии подчинено создание онтологий в рамках DOE, и на этих же принципах базируется весь инструмент, как его структура, так и интерфейс.

2.2. Protégé

Protégé является локальной свободно распространяемой программой, разработанная в Стэнфордском университете¹. Инструмент имеет графический интерфейс, удобный для использования неопытными пользователями, снабжен справками и примерами. Protégé основан на фреймовой модели представления знания ОКВС (Open Knowledge Base Connectivity) и снабжен рядом плагинов. В рамках Protégé выделяют Protégé Frames и Protégé OWL, каждый из них обладает специфическим интерфейсом. В

¹ *Gómez-Pérez A.* A survey on ontology tools. Ontoweb. Ontology-based information exchange for knowledge management and electronic commerce // URL: http://www.aifb.uni-karlsruhe.de/WBS/ysu/publications/OntoWeb_Del_1-3.pdf

данной работе мы затронем только Protégé Frames, и далее под Protégé будем подразумевать именно Protégé Frames.

Так, в Protégé присутствуют:

- классы;
- слоты (= свойства);
- экземпляры;
- фасеты (= ограничения слотов).

3. Специфика онторедкторов DOE и Protégé

Теперь, когда мы прояснили некоторые общие черты онторедкторов DOE и Protégé, продемонстрируем специфику сравниваемых редакторов, переходя от одного элемента онтологии к другому. Мы рассмотрим особенности DOE и Protégé в отношении 1) их повторного использования в других приложениях; 2) работы с классами; 3) представления свойств и отношений; 4) описания экземпляров; 5) использования аксиом.

3.1. Повторное использование DOE и Protégé в других приложениях

Одним из преимуществ онтологий считается возможность ее повторного использования в других приложениях. В связи с этим очень важным является простота ее переноса на другие платформы для внедрения в другие программные продукты и онтологии. Это подразумевает возможность перевода онтологий на другие формальные языки. Редактор DOE предоставляет возможность осуществлять импорт (RDF, OWL, импорт в собственный формат) и экспорт (экспорт из собственного формата, CGXML, DAML+OIL, OIL Plain Text, Oil XML, RDFC, OWL RDF/XML Exchange Syntax, OWL XML Presentation Syntax). Однако на данном этапе порой возникают проблемы, и экспорт не осуществляется по ряду технических причин, таких как, например, наличие специальных символов в онтологии, созданной в DOE, которые запрещены в других формальных языках. Также особен-

ностью DOE является то, что эта программа разрешает использование пробелов в именах элементов, хотя другие инструменты не всегда поддерживают эту возможность. Система Protégé предоставляет возможность импорта в собственные форматы (.pont and .pins), а также в RDF, RDFS и OWL. Экспорт осуществляется в форматы RDF; RDFS; XML; OWL; Clips; N3; TURTLE.

3.2. Работа с классами в DOE и Protégé

При сравнении онторедкторов выявляются фундаментальные отличия в работе с классами. Как уже упоминалось ранее, в основе методологии, используемой в DOE, лежит формулирование четырех отличительных свойств при описании классов. При этом *Similarity with Siblings* (сходство с братьями) и *Similarity with Parent* (сходство с отцом) задаются для всех братьев в иерархии, то есть при изменении их для одного понятия они изменяются и для координированных понятий. Так, например, в нашей иерархии присутствовало понятие *Расходы* и его видовые понятия *Условно-постоянные расходы* и *Переменные расходы*. Если мы изменяли *Similarity with Siblings* для понятия *Переменные расходы*, то автоматически изменения происходили и для понятия *Условно-постоянные расходы*, о чем программа предупредила. Однако *Similarity with parent* (сходство с отцом) нигде в явной форме не используется. Также сложно было понять разницу между *Similarity with Siblings* и *Similarity with Parent*. Такой подход порой помогает формализовать описание и найти основание для выделения вида из родового понятия, но порой представляется избыточным.

После того как заданы три дифференциальных признака, существует возможность автоматического формирования определения для понятия или так называемого *Difference with Parent* (отличие от отца). Оно складывается из *Similarity with Siblings* (сходство с братьями) и *Difference with Siblings* (отличия от братьев), представляемые через двоеточия. Однако генерируемые

определения избыточны, громоздки, их сложно сокращать. Так, можно привести определения, автоматически сгенерированного в рамках DOE для понятия *Инвестиционный риск: показатель, измеряющий варибельность дохода или доходности, когда предприятие характеризуется с позиции источников средств: варибельность дохода или доходности в результате процесса инвестиционной деятельности предприятия.*

С другой стороны, преимуществом интерфейса DOE является возможность отображения в определении всех вышестоящих понятий, что порой может быть удобным. Так, для понятия *Постоянные расходы* мы можем сразу увидеть определения для понятий *Расходы* и *Условно-постоянные расходы*, стоящих выше в иерархии. С точки зрения работы с классами также является удобным наличие уже встроенного поля «Синонимы», а также «Лексические предпочтения», где можно выбрать предпочтительное языковое выражение для данного понятия. Таким образом, в качестве имени класса можно выбрать одно наименование, а в качестве предпочтительного языкового выражения совершенно другое.

Стоит упомянуть, что при работе с DOE возникли некоторые трудности технического характера. Этот инструмент адаптирован лишь для английского, французского, немецкого и ряда других языков. Однако русский язык не вошел в этот список, таким образом, при создании онтологии пришлось использовать транслитерацию. Аналогично возникли сложности с употреблением специальных символов, таких как кавычки, присутствующие в названиях некоторых терминов. Данная проблема решалась их заменой символами нижнего подчеркивания («_»).

В онторедакторе Protégé существует возможность описания так называемых метаклассов. В иерархии метаклассов содержатся классы, слоты и фасеты. Метакласс представляет собой своего рода «шаблон» для описания элементов, создаваемых в рамках этого метакласса. Здесь можно, например, описывать, по каким

параметрам понятия, создаваемые в рамках данного метакласса, могут получать характеристики. Так, например, мы указываем, что понятия, создаваемые в рамках метакласса ITEMS могут иметь синонимы. Далее при создании класса мы получаем возможность заполнять поле «Синонимы», добавляя новые синонимы. Далее, работая с редактором форм и применяя его для соответствующего метакласса, можно выбрать тип представления классов на экране, например, можно указать, что в иерархии будут отображаться не сами понятия, а их прямые виды или же их определения. Следует заметить, что Protégé относится к системам представления знаний, которые позволяют осуществлять множественное наследование в иерархии классов, то есть класс может быть подклассом нескольких классов. Данная возможность в рамках DOE, например, отсутствует. Однако в Protégé, в отличие от DOE, изначально не предусмотрено добавление синонимов, и для этого нужно специально предусмотреть отдельный слот при описании метакласса в процессе создания онтологии.

Protégé свободно поддерживает различные языки, в том числе и русский, что значительно облегчает работу. В отличие от DOE, данный онторедатор запрещает использование пробелов в именах классов и свойств, в нашей онтологии пробелы были заменены символами нижнего подчеркивания («_»). Запрет на использование пробелов не позволял свободно вносить такие понятия, как *Норма прибыли*, *Премия за риск* и многие другие. Этот запрет можно отнести к неудобствам редактора Protégé.

3.3. Представление свойств и отношений в DOE и Protégé

Рассмотрим онторедаторы DOE и Protégé с точки зрения работы со свойствами и отношениями. В рамках DOE есть возможность строить иерархии отношений, таким образом проясняя специфику отношений, их взаимосвязь. Вместе с тем, иерархия отношений существует независимо от иерархии классов, то есть в дифференциальной части онтологии нет связи между понятиями

и отношениями, она появляется лишь в референциальной части, где можно определить домены отношения, указывая на принадлежность аргументов этих отношений к определенным классам. Однако в примерах онтологий, приведенных в DOE, домены порой вообще не описываются или задаются не до конца (не конкретизируется домен, но указывается, что это, например, бинарные отношения). Оказывается невозможным задание конкретных экземпляров отношений между, например, отдельными экземплярами понятий, которые задаются списком, связанным с каким-то понятием. Так, нам было необходимо указать, что понятие *Риск* включает два аспекта (рассматриваемые как меронимы): *Систематический риск* и *Несистематический риск*. В данном редакторе связать эти три понятия не представлялось возможным. В рамках DOE нет возможности описания универсальных отношений, которые могут применяться для понятий, относящихся к разнообразным классам (как, например, в случае партитивных отношений). Разумеется, такого рода отношения можно вводить, не задавая домены вообще, но тогда полностью утрачивается связь с иерархией понятий.

В Protégé также есть возможность построения иерархии отношений (называемых в Protégé слотами), но в рамках метаклассов. Любой слот является реализацией метакласса STANDARD_SLOT, включенного в иерархию метаклассов. Однако в Protégé нет ограничений на использование слотов для связи элементов онтологии, что делает онтологию более наглядной и структурированной. Есть возможность связать отдельные классы или экземпляры каким-то отношением. В рамках данной среды разработки можно свободно создавать и широко описывать новые отношения в дополнение к стандартным. Список таких стандартных слотов также велик, например: :CREATOR; :DIRECT DOMAIN; DOCUMENTATION; :ROLE; :NAME и другие. Они могут применяться в различных областях и являются универсальными.

Для нашей онтологии в дополнение к стандартным слотам мы создали ряд собственных: `:SYNONYM`; `:is_HOLOMYM_of`; `:is_MERONYM_of`; `:is_INSTRUMENT_of`; и некоторые другие. Это позволило в дальнейшем беспрепятственно задавать отношения меронимии и не только. Так для связи понятия *Риск* с его меронимами было достаточно указать, что *Риск* имеет два меронима. Также создание дополнительных слотов решило проблему отображения ассоциативных отношений, что открыло возможность для детализации онтологии.

В Protégé присутствуют также так называемые фасеты – своеобразные ограничения, накладываемые на слоты. К фасетам относятся мощность отношения, домен, диапазон и некоторые другие. Использование фасетов дает возможность конкретизировать отношения, описывать их более подробно, таким образом, в онтологию может быть включено больше информации.

Как упоминалось ранее, слоты могут связывать между собой классы или экземпляры, но также возможно установление отношений между самими слотами, например: `owl:equivalentProperty` (определяет эквивалентное свойство) и `owl:inverseOf` (определяет обратное свойство). Это упрощает работу со слотами, так как, например, при заполнении одного из слотов, состоящего в отношении `owl:inverseOf`, значение обратного ему заполняется автоматически. Так, например, в нашей онтологии обратными являлись `:is_Meronym_of` и `:is_HOLONYM_of`, отражающие партитивные отношения.

3.4. Описание экземпляров в DOE и Protégé

Работа с экземплярами в рамках данного исследования не велась, однако хочется упомянуть и этот компонент онтологии в обсуждаемых онторедаторах. Хотя референциальная часть DOE мало использовалась при создании нашей онтологии, был замечен ряд ее особенностей. Так, дифференциальная и референциальная онтология формируются одновременно, при добавлении в

дифференциальную часть элемент появляется автоматически и в референциальной. Аналогично дело обстоит и с удалением, но стоит упомянуть, что удалять элемент, занесенный в дифференциальной онтологии, в рамках референциальной нельзя. Однако система понятий дифференциальной онтологии может не полностью совпадать с системой понятий референциальной онтологии, это достигается за счет возможности занесения элемента в референциальной части, который не появляется в дифференциальной (этот элемент можно удалять в референциальной онтологии). Предназначение этой опции и ее применимость для практических целей не очень ясна. В референциальной части онтологии понятию можно приписать второго отца помимо того, что уже приписан в дифференциальной. При этом понятие в референциальной онтологии помечается особым образом, однако в дифференциальной онтологии это никак не отображается. В DOE легко добавлять новые элементы, но, как упоминалось ранее, нет возможности связывать их между собой какими-либо отношениями.

В Protégé работа с экземплярами также не представляет особой сложности, но существует намного больше возможностей, чем в DOE. Так, в Protégé можно приписывать какие-то конкретные характеристики экземпляру, задавать отношения между экземплярами. Существует также возможность установления связи между экземпляром и классом. В рамках Protégé можно определять форму представления экземпляров на экране, что, безусловно, является удобным и адаптирует интерфейс под конкретную задачу.

3.5. Аксиомы в DOE и Protégé

Говоря об аксиомах, стоит заметить, что в DOE они не поддерживаются, а в Protégé существуют как дополнительная опция. Однако в Protégé возможно использование сложных запросов для поиска в рамках онтологии, то есть можно осуществ-

для выборки информации из созданной структуры. Пример такого запроса представлен на рис. 1:

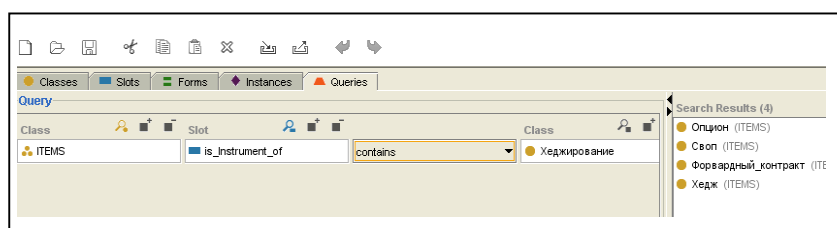


Рис. 1. Запрос в системе Protégé

Этот запрос ищет все элементы, которые являются инструментом «хеджирования». Результат данного поиска отображается в правой части, где указано, что инструментом *Хеджирования* считается *Опцион*, *Своп*, *Форвардный контракт* и *Хедж*.

Таким образом, можно осуществлять проверку информации при разработке онтологии, а также использовать созданную онтологию в качестве упрощенной вопросно-ответной системы.

4. Выводы

Подводя итог, можно сказать, что DOE проще в использовании, а Protégé предоставляет больше возможностей для адаптации под конкретную задачу и конкретного пользователя. В рамках нашего исследования предпочтение было отдано Protégé как системе, дающей возможность эффективно отображать связь элементов друг с другом.