УДК 519.7:007.52

РАСШИРЕНИЕ ВОЗМОЖНОСТЕЙ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ПРИ ИСПОЛЬЗОВАНИИ БАЗ ЗНАНИЙ, ОСНОВАННЫХ НА OWL 2.0

Д.А. Плиско*
Контактный тел.: 091-905-99-42
E-mail: PliskoDmitry@gmail.com

А.Ю. Шевченко
Кандидат технических наук, преподаватель*
*Кафедра искусственного интеллекта
Харьковский национальный университет радиоэлектроники
пр. Ленина, 14, г. Харьков, Украина
Контактный тел.: 050-619-26-38
E-mail: shevchenko@sw-expert.com

У статті розглянуті переваги використання онтологічних баз знань, що відповідають специфікації OWL 2.0. Розглянуті відмінності від попередніх стандартів, що найбільш повно демонструють відмінності OWL 2.0

Ключові слова: онтологія, OWL, логіка, висновок

В статье рассмотрены преимущества использования онтологических баз знаний, соответствующих спецификации OWL 2.0. Рассмотрены отличия от предыдущих стандартов, которые наиболее полно демонстрируют отличия OWL 2.0

Ключевые слова: онтология, OWL, логика, вывод

In this article examined advantages of the ontology-based knowledge bases using. Examined differences from previous standards, that most fully demonstrate distinctions of OWL 2.0

Key words: ontology, OWL, logic, reasoning

1. Введение

С первых дней развития Semantic Web появилась необходимость в языке описания онтологий. Первые исследования были ориентированы на языки RDF и RDF Schema, однако их выразительной мощности не хватало. В связи с этим, Консорциум W3C сформировал Рабочую группу онтологий Web (Web Ontology Working Group), целью которой стала разработка языка описания онтологий для использования в Semantic Web. Таким языком стал OWL (Ontology Web Language). С февраля 2004 года OWL стал рекомендацией W3C.

По сути OWL представляет из себя семейство языков, содержащее в себе 3 варианта:

1) OWL Lite – простейшая редакция языка, содержащая в себе простейшие инструменты описания классификационных иерархий и ряд ограничений, что позволило получить функциональный язык при сравнительно формальной сложности, что позволяет значительно ускорить обработку документов.

2) OWL DL – редакция языка, предоставляющая максимальную выразительную мощность при сохранении вычислимости – все высказывания, удовлетворяющие требованиям OWL DL будут гарантированно вычислимыми.

3) OWL Full – редакция, предоставляющая максимальную выразительную мощность без сохранения гарантированной вычислимости результатов. По сути ограничениями OWL Full являются синтаксические ограничения языка RDF.

Результатом разработки и стандартизации OWL стала разработка и адаптация ряда программных продуктов, использование онтологий в различных науках: медицине, биологии, географии, автомобильной и космической промышленности.

Однако, не смотря на множество удачных примеров применения OWL, при использовании в нем были обнаружены различные недостатки. С целью их устранения, а так же улучшения языка были внесены незначительные изменения. Таким образом, в 2005 году было достигнуто соглашение по введению новой редакции - OWL 1.1.

Внесенные изменения не остановили прогрессивного развития языка, и в сентябре 2007 была сформирована официальная рабочая группа Консорциума W3C, целью которой стала модернизация OWL, «очистка» спецификации языка, а так же его модифи-

кация с целью создания стабильной платформы для последующих разработок. В апреле 2008 года было принято решение назвать язык OWL 2, что отображает новый этап в его эволюции.

Основными изменениями, внесенными в новый язык стали:

1) Расширение выразительной мощности языка при сохранении его вычислимости.

2) Устранение ряда синтаксических недостатков, а так же внесение «синтаксического сахара».

3) Расширение возможностей метамоделирования.

4) Улучшение механизма импорта и контроля версий.

5) Изменения в аннотациях.

6) Введение профилей OWL 2 EL, OWL 2 QL и OWL 2 RL.

Итогом работы стало получение OWL 2 статуса W3C Proposed Recommendation 22 сентября 2009 года. В наиболее яркими отличиями OWL от OWL 2.0 являются такие важные элементы как: выразительная мощнность языка и механизм аннотаций.

2. Выразительная мощность

Опыт практического применения OWL DL показал, что, несмотря на его высокую выразительную мощность, все-таки не хватало ряда конструкций для моделирования сложных доменов. При использовании OWL, решением данной проблемы было создание искусственных шаблонов, близких к логике необходимых высказываний, однако при этом, как правило, получались довольно громоздкие конструкции, которые сложно изменить, и требующие высокой вычислительной мощности при обработке.

К таким ограничениям, в первую очередь, можно отнести ограничения по количеству элементов (Qualified Cardinality Restriction, QCR). Под ним подразумевается описание класса, объекты которого содержат точное количество определенных составляющих. В OWL данный недостаток разрешался использованием различных шаблонов, однако конструкции, полученные в результате их применения, были как правило громоздкими, неочевидными и неполными. Уже в ходе разработки OWL была возможность добавить QCR к языку без значительного прироста в сложности вычислений, однако первая реализация все же появилась лишь в DAML+OIL – языке, построенном на базе OWL.

3. Относительная выразительность

Сюда следует отнести такие понятия, как «распространение относительно свойств» и «свойства свойств».

Распространение относительно свойств (Propagation along properties) подразумевает, что свойство части может быть свойством целого, а может и не быть. Например, поломка части механизма означает поломку механизма, однако поломка одной детали в партии не означает поломку партии. Для решения задач с подобными механизмами существуют специальные языки, такие как OBO и SNOMED, однако они не имеют мощности описания иерархий классов как OWL.

Свойства свойств (properties of properties). Как правило, отношение partOf – часть, определяется как транзитивное, рефлексивное и ассиметричное. Однако в ряде задач существует необходимость сохранения лишь некоторых из этих трех свойств (например, отношения транзитивные и рефлексивные, транзитивные и иррефлексивные или только иррефлексивные). Во время разработки OWL еще не было окончательного мнения о том, можно ли расширить язык свойствами с изменяемыми свойствами, поэтому в первую редакцию (OWL 1) вошло только транзитивное, рефлексивное и ассиметричное свойство partOf. Однако как и в случае с QCR на практике часто возникала необходимость в таких решениях, и, как правило, решение было таким же – применение громоздких, неочевидных и неполных шаблонов.

Решением проблем с выразительной мощностью стал пересмотр аппарата дескриптивной логики SHOIN. После глубокой переработки в OWL 2 был реализован аппарат, имеющий значительно более широкие выразительные свойства, чем SHOIN, сохраняя при этом вычислимость OWL DL. Этот аппарат получил название SROIQ. В него, в частности, вошли возможности QCR, новые возможности по работе со свойствами: аксиомы включения сложных свойств, управление транзитивностью, рефлексивностью и симметричностью свойств.

При всех нововведениях, вычислительная сложность SROIQ составляет 2NExpTime, в то время как вычислительная сложность SHOIN NExpTime. В то же время, так как SROIQ основан на SHOIN, если не использовать нововведения то вычислительная сложность NExpTime сохраняется.

4. Ограничения по типам данных

Язык OWL сильно ограничен в типах данных. Например, в OWL нельзя определить следующие выражения:

1) Примитивный тип с ограничениями по диапазону (например, строка длиной от 3 до 10 символов, строка из букв от «a» до «f» и т д.).

2) Взаимодействие между свойствами объектов (например, квадрат – если ширина равна длине).

3) Взаимодействие между различными объектами (например, люди, которые старше своего начальника).

4) Агрегационные функции (например, периметр фигуры равен сумме длин ее сторон).

Кроме того, не смотря на то, что OWL базируется на XML Schema, в OWL необходима реализация только типов xsd:string и xsd:integer. Остальные же типы используются опционально. Это связано с тем, что обработка типов в OWL и XML Schema проводится по разному, в частности, для описания эквивалентности конечных примитивных типов xsd:double и xsd:float требуется довольно большая конструкцией дизъюнкций.

Что же касается OWL 2, то множество встроенных типов данных было значительно расширено: OWL 2 полностью поддерживает такие типы данных как owl: boolean, owl:string, xsd:integer, xsd:dateTime, xsd:hexBinary, а так же набор типов, получаемых из них базовых введением ограничений. Кроме того, появилась возможность трансформации xsd:decimal, xsd:double и xsd:float в owl:

real. Такая возможность обеспечивает удобный переход между типами XML Schema и типами OWL.

Важным нововведением стало также появление ограничений по типам данных (DatatypeRestriction). Это позволит решить часть описанных выше задач по использованию типов данных.

Следует заметить, что Рабочая группа OWL не пришла еще к единому окончательному мнению относительно множества типов и механизмов их взаимодействие, поэтому следует рассчитывать на нововведения и изменения в будущем.

5. Ключи

OWL DL не предоставляет механизмов определения ключей, что особенно необходимо ввиду развития технологии баз данных. Например, в OWL DL не существует базовых конструкций для описания понятия «данный объект однозначно определяется этим свойством». В принципе, такая конструкция реализуема в OWL Full, однако отсутствие гарантированной вычислимости OWL Full делает такую возможность значительно менее привлекательной. Кроме того, во всех вариантах OWL отсутствует возможность построения составных ключей, понятия, определенного в терминах реляционных баз данных и часто являющегося необходимым для построения моделей сложных информационных структур.

Несмотря на значительно возрастающую вычислительную сложность конструкций с ключами, следует отметить, что такие конструкции являются крайне необходимыми для множества приложений. Как показали практические попытки добавления механизма ключей, их использование создает ряд теоретических и практических проблем. Поэтому в OWL 2 было решено внести механизм так называемых «простых ключей» («easy keys»). Суть механизма заключается в ведении ключевой аксиомы HasKey(), содержащей множество ключей P, причем такое, что никаких 2 объекта не могут иметь одинаковую совокупность этих ключей. Таким образом, в высказываниях можно использовать это свойство для описание связи с объектом.

6. Синтаксические изменения

При разработке OWL были определены 2 нормативных синтаксиса: абстрактный синтаксис (Abstract Syntax) и OWL RDF Syntax. Рекомендации W3C включают также XML синтаксис, однако он не является нормативным, и, следовательно, не получил широкого распространения. Абстрактный синтаксис является базовым оригинальным описанием языка и послужил основой для разработки API (интерфейса для разработчиков, Application Programming Interface). Однако следует заметить, что оба синтаксиса имеют ряд неточностей, что создает трудности для синтаксического разбора и создает ряд ошибок при трансформации онтологий, описанных одним синтаксисом, в другой.

К сложностям синтаксиса в первую очередь следует отнести различия в подходах и интерпретации конструкций. С одной стороны, OWL ничем не противоречит фреймовой парадигме и определение любого класса по сути, может рассматриваться как определение фрейма. Однако, с другой стороны, в основе OWL лежит дескриптивная логика, оперирующая не фреймами, а аксиомами. Именно из за таких несоответствий в подходах возможны различные варианты интерпретации, и в результаты для фреймовой и логической могут оказаться разными.

Также проблемой является интерпретация различными средствами обработки, реализуемыми различными разработчиками. С учетом внутренних преобразований каждой из систем обработки до внутреннего состояния, возможны незначительные изменения в логической интерпретации, количество которых будет увеличиваться с каждой последующей интерпретацией.

Помимо этого, в языке OWL нет логической типизации, то есть разделения словаря на объекты, классы, свойства объектов и данные. Это значительно увеличивает свободу построения синтаксических конструкций, однако и создает ряд проблем при синтаксическом разборе. Часто технически трудно разобрать, какой тип в данном случае используется и, следовательно, результаты интерпретации будут зависеть в первую очередь от технической реализации средств обработки. Эта проблема, совместно с описанной выше проблемой внесения изменений при обработке каждым из средств порождает еще больше неточностей при преобразованиях.

Также следует заметить, что нечеткая типизация не дает технической возможности контроля ошибок разработчиков, то есть нет возможности для отслеживания опечаток или ошибочного использовании понятий в ходе разработки. Часть таких ошибки могут быть выявлены только на этапе тестирования разработанной онтологии, однако, как показывает практика применения языков программирования со строгой типизацией (таких как C++, Java), возможно лишь уменьшить количество ошибок, однако не исключить их полностью.

Частичным решением этой проблемы стало использование в OWL 2 механизма объявляния (declaration). Все сущности OWL 2 могут, а иногда и обязаны, быть объявлены. Механизм объявлений нужен для определения сущности как класса, типа данных, свойства объекта, свойств-данных и свойств-аннотаций, объектов. Это позволяет обеспечить однозначность при использовании сущностей. Помимо этого введен механизм контроля последовательности объявлений, то есть все используемые сущности должны быть ранее объявлены. Это не накладывает значительных ограничений на структуру онтологий и не является обязательным, однако позволяет отследить ряд простых ошибок.

Что касается RDF варианта синтаксиса, то к основным недостаткам следует отнести довольно высокую сложность описания понятий. Зачастую, простейшие конструкции, которые просто записываются с помощью абстрактного синтаксиса, а RDF варианте представляют собой большое количество триплетов, что делает такие онтологии практически неудобными для чтения человеком. Также важно, что официальная спецификация описывает только соответствие между синтаксисами, однако не описывает порядок трансформации. Поэтому опять же существует возможно-

сти возникновения неточностей, которые зависят от конкретных инструментов преобразования и могут приводить к искажению оригинальной онтологии.

Нововведениями в области синтаксиса для OWL 2 стали так же стандартизированная поддержка XML синтаксиса в качестве альтернативы RDF. Это сделано с целью упрощения синтаксического разбора онтологий, опубликованных в сети интернет, обеспечении более удобного и гибкого взаимодействия с существующими протоколами сети.

Дополнительным нововведением также стала поддержка функционального синтаксиса (Functional-Style Syntax) с целью замены устаревшего абстрактного (Abstract) синтаксиса. Новый синтаксис стал более вербализированным, более удобным для чтения человеком, более удобным для публикации материалов в сети Интернет. Важно заметить, что не была реализована обратная поддержка абстрактного синтаксиса, то есть онтология, написанная на абстрактном синтаксисе, не будет состоятельной и правильной онтологией функционального. Таким образом, использование онтологий OWL стало бы невозможным. Решением данной проблемы стала неизменность синтаксиса RDF, то есть, с одной стороны, нововведения OWL 2 никак не отразились на RDF синтаксисе, с другой стороны, любая онтология, написанная на RDF, является правильной онтологией OWL 2. Таким образом, были разработаны механизмы трансляции онтологий из функционального синтаксиса в RDF и обратно без изменений в логике онтологии. С учетом того, что существует совместимость абстрактного OWL синтаксиса и RDF, то в целом можно считать, что задача обратной совместимости решена.

7. Метамоделирование

В ряде практических задач возникает проблема разделения понятия класса и объекта класса. Часто используется одно понятие, которое в зависимости от контекста может означать класс, группу объектов или отдельно взятый объект. Такой подход называется метамоделированием.

Во время разработки OWL метамоделирование не было распространено и, следовательно, необходимость в таком инструменте появилась позднее. Поэтому в OWL не вошли инструменты метамоделирования (за исключением OWL Full, где они реализованы частично). Но следует заметить, что данный подход является довольно удобным при решении разного рода задач и существует необходимость в поддержке данного подхода.

Для решения задач метамоделирования в OWL 2 хорошо известный язык метамоделирования MOF (Meta-Object Facility). Классы модели MOF описывают каноническую структуру онтологии в виде набора аксиом и аннотаций, отношений между ними а так же отношения структурной эквивалентности, что упрощает совместное использование онтологий а так же обработку их существующими инструментами.

В целом, MOF можно рассматривать как аналог DOM модели для XML. Она определяет ряд понятий и структуру онтологий, что необходимо для разработки инструментов и интерфейсов программирования.

8. Импорт и контроль версий

Язык OWL предоставляет следующий механизм импорта онтологий: каждая импортируемая онтология определяется по двум параметрам – имени онтологии и ее размещению. Такой подход позволяет физически разделить онтологии, однако при этом множество аксиом разрабатываемой и импортированных онтологий рассматриваются как единое целое. Следует отметить, что такой подход удобен при постоянном размещении онтологий и неизменности их имен, но на практике часто возникает необходимость в изменении местонахождения и имени. При этом онтологии, импортировавшие данную, не могут ее больше использовать. Существующие на данный момент решения, такие как кэширование внешних онтологий не могут полностью решить эту проблему. Что касается контроля версий – так же трудно определить, как сохранять различные версии и какую из версий необходимо использовать.

Принципы публикации, импорта и контроля версий в OWL 2 несколько отличаются. В частности, в OWL 2 введен механизм перенаправления (redirect). Таким образом, при физическом переносе онтологии сохраняется работоспособность тех онтологий, которые ее импортируют. Что же касается контроля версий, то в OWL 2 введен достаточно простой принцип управления версиями. Теперь URI может определять как одну конкретную версию онтологии, так и так называемую серию онтологий (ontology series), то есть набор версий одной и той же онтологии. Новая версия добавляется в серию, при этом с помощью старого URI можно использовать любую из версий.

9. Механизм аннотаций

Механизм аннотаций OWL базируется на элементах RDF Schema и включает в себя следующие элементы: owl:versionInfo, rdfs:label, rdfs:comment, rdfs:seeAlso, rdfs:isDefinedBy. Возможны так же собственные элементы аннотирования, однако существуют проблемы в их использовании в OWL DL: это связанно с тем, что OWL DL запрещает использование аннотаций в аксиомах с целью реализации их экстралогических свойств – изменения в аннотациях не должны влиять на результаты обработки онтологии.

Как правило, большинство инструментов просто игнорируют аннотации. Такие недостатки не дают аннотировать аксиомы, а так же невозможно определять домены и наборы иерархий свойств аннотаций, что часто бывает необходимо в практических задачах.

В OWL 2 появилась возможность аннотировать так же и аксиомы. При этом аннотации не влияют на семантику онтологии, но при этом в целом сохраняется структура аннотирования.

10. Профили

Как и в OWL, в OWL 2 существуют «урезанные» версии языка (иногда называемые «фрагменты» или «субъязыки»), однако, в отличии от OWL, структура несколько изменилась – были созданы не 3 редакции языка, а язык OWL 2, который является самым пол-

ным, и 3 «урезанные» версии для решения конкретных задач: OWL 2 EL, OWL 2 QL и OWL 2 QL. Каждая из версий предоставляет различную степень выразительной мощности и предназначена для различных целей.

OWL 2 EL базируется на семействе EL++ логик, предназначенных для эффективного вывода в задачах с обширной терминологией, в частности задач классификации. При этом вычислительная сложность решения таких задач является полиномиальной. В данной версии языка используется только конъюнкция и квантор существования (SomeValuesFrom). Такой подход позволил получить инструмент с возможностью быстрого вычисления для определенного круга задач.

Более мощным инструментом является профиль OWL 2 QL, который базируется на DL-семействе дескриптивных логик. В данном случае целью является разрешения конъюнктивного запроса, а в описании можно использовать так же дизъюнкцию и квантор всеобщности (AllValuesFrom).

Еще более полным является OWL 2 RL. Он включает почти все конструкции OWL 2 и поддерживает вывод основанный на правилах, поддержку импликаций первого порядка и анонимных объектов. Основой для OWL 2 RL стали идеи дескриптивного логического программирования (Description Logic Programming).

11. Вывод

В статье проведен анализ стандартов OWL и OWL 2.0, выявлены все особенности связанные с использованием стандарта OWL 2.0. Описаны его основные концепции и методы, что позволит в значительной степени усовершенствовать метод построения информационных интеллектуальных систем на основе онтологических баз знаний. Рассмотрены тенденции развития онтологических интеллектуальных баз знаний, выявлены их основные приоритеты развития это и разработка макросредств для ускорения и упрощения разработки, различных языков запросов, интеграции с RIF (Rules Interchange Format) и возможностей немонотонного вывода данных.

Литература

1. B. C. Grau, I. Horrocks, B. Motik, B. Parsia, P. Patel-Schneider, U. Sattler. OWL2: The Next Step forOWL. (2008).
2. S. Bechhofer. Parsing OWL in RDF/XML, W3C Working Group Note (2004).
3. I. Horrocks, P. F. Patel-Schneider, Reducing OWL Entailment to Description Logic Satis ability// Journal of Web Semantics, 2004.–№1 (4).

У статті було розглянуто фільтр CAPTCHA з точки зору його ефективності та простоти використання. На основі цього була запропонована інформативна схема тестування фільтра для перевірки його якості

Ключові слова: фільтр, захист інформації, CAPTHCA

В статье был рассмотрен фильтр CAPTCHA с точки зрения его эффективности и простоты использования. На основе этого была предложена информативная схема тестирования фильтра для проверки его качества

Ключевые слова: фильтр, защита информации, CAPTHCA

This article was reviewed CAPTCHA filter in terms of its effectiveness and usability. On the basis of this scheme was proposed informative structure testing to verify its quality

Key words: filter, protected of information, CAPCHA

УДК 681.3

АНАЛІЗ І ТЕСТУВАННЯ ЯКОСТІ CAPTCHA

І.С. Прись
Харківський національний університет радіоелектроніки
просп. Леніна, 14, м.Харків, Україна, 61166
Контактный тел.: 093-946-80-55
E-mail: johniman@mail.ru

Вступ і постановка задачі

Розміщення CAPTCHA призначене для того, щоб захистити сайт від спаму, щоб відрізнити людину від комп'ютера, або, в даному випадку, бота. CAPTCHA - це програма яка генерує і оцінює тести, які можуть бути вирішені людиною, але недоступні для існуючих на сьогоднішній день комп'ютерних програм [2].

Таким чином, добра якісна CAPTCHA повинна бути одночасно надійною і легкою для сприйняття людини.

CAPTCHA — це програма для того, щоб запобігати численні автоматичні реєстрації та відправлення