Спецификации как онтологии

авторы: Манцивода А.В., Стукушин Н.О.

Формализованные спецификации используются во многих областях. Они полезны при решении множества важных задач, например, при разработке опытных образцов систем, компиляторов и систем метаданных. Вероятно, самый известный пример – использование контекстно свободных грамматик как формальных описаний языков программирования для автоматической генерации синтаксических анализаторов.

Формализация сравнительно легка для тех стандартов и спецификаций, которые описывают протоколы обмена данными, метаданные или спецификации, связанные с конкретными техническими решениями. Но если спецификация многокомпонентна и ее значительная часть посвящена концептуальным вопросам, для формализации необходима выразительная логическая система, способная описывать разноуровневые компоненты спецификации. Такой формализм существует (например, логика первого порядка), но не очень помогает, поскольку слишком сложен в вычислительном отношении (вплоть до неразрешимости), чтобы успешно использоваться на практике. Если формальная система сложна, формализация, основанная на ней, не будет работать на реальных задачах. Таким образом, существует необходимость в некотором компромиссе между выразительностью формальной системы и ее простотой и обозримостью. Кроме того, такая формальная система должна быть достаточно универсальной и гибкой для того, чтобы допускать настройку в соответствии с особенностями формализуемой спецификации.

В данной работе рассматривается идея использования дескриптивных логик [1] и онтологий для формализации спецификаций. Дескриптивные логики являются универсальным инструментом для формального описания знаний. Они предоставляют богатые выразительные средства, которые позволяют включать формализованные спецификации в автоматические системы знаний. Диапазон различных дескриптивных логик весьма широк, что позволяет выбирать из них наиболее подходящие для целей формализма. Благодаря стандартизированному представлению онтологий в виде языка OWL (Web Ontology Language, [2]), имеющего строгую логическую семантику, формальное описание спецификации в виде онтологии обеспечивает однозначное понимание концептов и конструкций спецификации. Также обеспечивается их переносимость и возможность автоматической обработки информации, связанной со спецификацией. Кроме того, формализация предоставляет большие возможности для анализа правильности спецификации и ее структуры, а также ее соответствия поставленным задачам.

Однако есть другой, абсолютно практический способ использования спецификаций, формализованных в виде онтологий. Ряд диалектов дескриптивных логик (включая языки OWL Lite и OWL DL) можно использовать, а некоторые, более простые, активно применяются. Поэтому онтологические спецификации могут напрямую использоваться как инструментальные средства для реализации информационных систем, работающих в той среде, которую описывает спецификация. Авторы хотят показать, что эти формальные описания

могут работать как инструменты прямого действия в качестве ядра таких информационных систем.

В этих системах онтологии играют роль продвинутых и интеллектуальных хранилищ данных. Использование онтологий как практических инструментов, ориентированных на управление данными, имеет ряд преимуществ, прежде всего полезных для самой семантической паутины. Во-первых, конкретные данные становятся полноценными участниками среды семантического веба. Оба уровня – терминологический (то есть уровень общих знаний) и уровень утверждений (уровень конкретных знаний об объектах) – объединяются не только теоретически, но и на практике. С концептуальной точки зрения это имеет существенное значение, поскольку возникает возможность сгладить дисбаланс, наметившийся в онтологиях, которые значительно более успешно используются для описания общих знаний о предметных областях, чем для описания конкретных данных о конкретных объектах, входящих в эти предметные области. Во-вторых, эффективное управление конкретными данными существенно для продвижения семантической паутины в обычное веб-программирование, а это крайне важно для будущего SW как новой парадигмы WWW. В-третьих, онтологии могут предложить красивые и выразительные методы управления данными, недоступные сегодня из-за неэффективности автоматических решателей, ориентированных главным образом на работу с алгоритмически тяжелыми терминологическими знаниями.

В данной статье рассматривается подход к формализации спецификаций на основе онтологий. Эффективная работа с объектами онтологии позволит использовать онтологическое представление спецификации как ядро информационной системы, которая использует конструкции спецификации как структуры данных. Таким образом, абстрактное и формальное описания имеют практическое применение. В качестве эталонного теста для экспериментов была выбрана широко известная спецификация консорциума IMS, Question and Test Interoperability (IMS QTI), описывающая формат для обмена и использования образовательных тестов и вопросов. Для манипуляций с объектами применялась программная оболочка OntoBox [3], которая базируется на простой, но очень эффективно реализуемой дескриптивной логике, моделирующей объектно-ориентированный подход.

Обработка данных в онтологиях

Для того чтобы онтологии стали практическим инструментом управления данными при решении практических задач, необходимо уметь эффективно работать с конкретной информацией (данными и объектами), хранящейся в них. В настоящее время онтологии наиболее часто и успешно применяются для описания общих (терминологических) знаний о предметных областях (например, в области биологии и медицины). К сожалению, использование онтологий как структур, ориентированных на хранение конкретных данных, удобных для реальных информационных систем, значительно отстает по популярности. Более того, широко распространено мнение, что онтологии менее удобны и эффективны на уровне конкретных объектов и для решения обычных задач, например веб-программирования, чем на уровне общих концепций. Во-первых, логическая основа онтологий слишком сложна для обычных разработчиков. Во-вторых, существующие сегодня методы работы с онтологиями совершенно неэффективны на уровне конкретных данных, поскольку настроены прежде всего на работу со сложными терминологическими знаниями. В-третьих, считается, что парадигма открытого мира онтологий входит в противоречие с

парадигмой закрытого мира, на которой построены обычные методы хранения данных (например в БД). К сожалению, эти причины, во многом надуманные, мешают распространению принципов SW во всемирной паутине. Фактически, если эти возражения верны, следует их принять, потому что более 95 % информационного наполнения WWW состоит из конкретных данных, опубликованных обычными разработчиками. Очевидно, если не сделать обычного веб-разработчика основной целевой группой SW, не удастся обеспечить существенное влияние SW на WWW. Авторы полагают, что первая проблема вполне преодолима, тогда как два последних постулата являются спорными и не соответствуют реальной практи- ке [3].

Для эффективной работы на уровне объектов и конкретных данных следует что-то сделать с автоматическими решателями, работающими в логической среде. Известно, что каждая разрешающая процедура является компромиссом между выразительностью и эффективностью, то есть существующие решатели, ориентированные на работу в сложной терминологической среде, слишком тяжелы для легкой, но быстрой работы с объектами. Очевидно, что эти методики нужно заменить более простыми, но в то же время более эффективными. Кроме того, просто выбрать новую методику для работы в логической среде недостаточно. Необходимо обеспечить среду эффективными инструментами для управления объектами онтологий, которые 1) конкурентоспособны по сравнению с обычными методами управления данными/объектами; 2) имеют логически непротиворечивую семантику, совместимую со стандартной семантикой дескриптивных логик; 3) поддерживаются специальным языком запросов, играющим роль, аналогичную роли SQL в системах управ- ления БД (в частности, это необходимо для то- го, чтобы спрятать от пользователя сложные и непопулярные конструкции математической ло- гики).

В [4] вводится понятие логической архитектуры, на основе которой формируется общая схема построения инструментов такого рода. Для этого основная логика L, в рамках которой описываются предметные области, стратифицируется на ряд уровней L1…Ln, где каждая страта ориентирована на решение определенной подзадачи. При этом каждая подлогика Li обеспечивается собственным пользовательским интерфейсом UIi и специальным пакетом модулей Modi (как логических, так и процедурных), ориентированных на обработку данных, описываемых в рамках страты Li. Логическая архитектура логики L – это коллекция компонентов Ci=áLi, UIi, Modiñ. Ключевая роль отводится базовой компоненте C0, которая ориентирована на работу с конкретными объектами онтологий. Очевидно, что C0 должна быть основана на простейшей логике, допускающей самые быстрые реализации. В [4] в роли такой логики предлагается модификация логики FL0 [1], расширенной типами данных. Эта модификация FL0 может быть охарактеризована как объектно-ориентированная логика (OO-логика). Интерфейс UI0 основан на языке запросов OntoBox QL [4] (фактически язык запросов основан на дескриптивной логике SHION(D) и может работать над любым уровнем логической архитектуры). Авторы полагают, что моделирование объектно-ориентированного подхода в рамках дескриптивной логики является очень важным шагом, поскольку обеспечивает среду, хорошо знакомую обычному веб-разработчику, что может упростить процесс вовлечения обычного разработчика в среду SW.

Данный подход теоретически обосновывается тем, что, как показано ранее, семантика OO-логики подобна системе структурных типов объектно-ориентированного языка программирования с множественным наследованием, и, таким образом, объекты онтологий могут быть отображены в объекты языка

объектно-ориентированного программирования.

Вернемся к спецификациям. Понятно, что конкретные конструкции, описываемые в спецификациях, должны моделироваться объектами соответствующей онтологии. Методы формального описания конструкций, используемые в рамках спецификаций, очень разнообразны. Некоторые описания основаны на специализированных языках спецификаций, какие-то используют объектные модели и т.д. Спецификация IMS для этой цели использует так называемую XML-привязку (XML binding). Недостаток XML заключается в том, что он может описать только структуру объектов, но не их семантику в контексте всей спецификации (для этого используется английский язык). В этом значительное отличие от онтологий, которые могут формально описать и общую семантику спецификации, и структуру ее конкретных конструкций. В онтологиях эта структура определена в общем контексте терминологического описания. Таким образом, XML-привязка может быть заменена на онтологическую привязку, которая в рамках одного языка формально описывает не только структуру, но и семантику этих конструкций. Вместе с общим описанием спецификации на терминологическом уровне формализация конкретных конструкций спецификации может служить и для реализации информационных систем, базирующихся на данной спецификации.

Спецификация IMS QTI

В этой работе рассматривается формализация спецификаций международного образовательного консорциума IMS [5]. Они включают в себя спецификации IMS Meta-data, Content Packaging и Question and Test Interoperability (версия 2.0). Основной в данном случае является IMS QTI, но она зависит от спецификаций IMS MD и IMS CP, поэтому они также должны быть формализованы. Каждая из спецификаций содержит описание своих основных конструкций в виде XML (XML-привязка). XML-привязка формально описывает объектную модель, которая сосредоточена на синтаксических и структурных аспектах, но не затрагивает вопросы семантики. Семантика этих конструкций описана отдельно на естественном языке (английском).

Задача состоит в том, чтобы описать каждую из трех спецификаций IMS в формате онтологий. Каждая спецификация представляется в виде отдельной онтологии. Эта онтология будет содержать только общее терминологическое знание о спецификации. В системах тестирования каждый конкретный тест представляется в отдельной онтологии, включающей объекты с конкретными данными о вопросах и ответах в рамках теста. Общие знания о поведении теста унаследованы этой конкретной онтологией от основных терминологических онтологий. Каждая отдельная онтология, содержащая тест, может свободно распространяться и повторно использоваться в разных контекстах, поскольку ее семантика и структура зависят только от терминологической онтологии IMS и не зависят от особенностей конкретных систем тестирования.

Онтологии, содержащие тесты, корректно поняты всеми сообществами, которые принимают терминологические онтологии IMS. Кроме того, если можно эффективно работать с объектами онтологии и система тестирования будет использовать онтологии для хранения данных, которые она обрабатывает, то онтология образовательных тестов может использоваться не только для распространения, но и для внутреннего представления информации в системе тестирования.

Технология реализации

Рассмотрим основные стадии разработки информационной системы, в которой онтологии используются как интеллектуальные инструменты хранения данных. На базе системы OntoBox проводились эксперименты со многими реальными задачами, для которых использование онтологий (вместе с эффективной обработкой объектов в OntoBox) представлялось весьма перспективным и выгодным. В результате была предложена общая схема разработки таких приложений. По существу эта схема включает четыре стадии. Предположим, что K – область знаний, в которой должно быть разработано приложение A.

Стадия A. Разрабатывается формализация используемых компонентов K в форме терминологических онтологий. Практика показывает, что почти всегда для подобных целей достаточно выразительности дескриптивной логики SHOIN(D) (которая соответствует OWL DL).

Стадия B. Части этих онтологий, определяющие объектную подмодель описания предметной области, загружаются в OntoBox – специальную программную среду, поддерживающую эффективную обработку объектов в онтологиях. OntoBox играет роль модуля памяти, в котором хранятся внутренние данные A.

Стадия C. Разрабатывается пакет методов обработки и моделирования поведения объектов в K. Так как методы, разработанные на данном этапе, определяют сущность предметной области K, они могут использоваться в любом приложении, относящемся к данной предметной области. Это означает, что они также могут повторно использоваться (вместе с онтологиями) и распространяться, например, в форме библиотек. Такой подход напоминает объектно-ориентированный, в котором структуры и методы работы с объектами включаются в описание класса. В частности, онтологии вместе с этими методами обеспечивают общую среду для модулей приложения A, среда же определяет компоненты логической архитектуры, соответствующей домену K.

Стадия D. Разрабатываются специальные модули в рамках среды онтологий, разработанной на стадии C. Данные модули призваны реализовать основные функции A. Система A формируется как единое целое.

Рассмотрим применение этой схемы в реализации системы тестирования, основанной на спецификациях IMS.

Стадия A: Формализация

Формальные описания трех спецификаций IMS были разработаны в рамках дескриптивной логики SHOIN(D) и представлены как онтологии.

Формализация спецификации QTI по существу содержит два уровня: уровень концептуального описания QTI и объектную модель QTI, которая описывает структуру вопросов, тестов и т.д. Для поставленной задачи более интересен второй уровень, описывающий конкретное представление тестов в системе проверки знаний. Объектная модель этого уровня может быть переведена в термы дескриптивной логики, которые будут непосредственно использоваться для работы с тестами как с объектами онтологии. Продемонстрируем на примере определение понятия теста, одного из базовых понятий спецификации QTI. Здесь (=n)R и (n..m)R – краткие обозначения ограничений на кардинальность $\geq n.R \sqcap \leq n.R$ и $\geq n.R \sqcap \leq m.R$ соответственно.

assessmentItem ϕ
(= 1) identifier ∏ (= 1) title ∏(0..1) label ∏
(0 :: 1) lang ∏ (= 1) adaptive ∏ (= 1) timeDependent ∏
(0..1) toolName ∏ (0..1) toolVersion ∏ responseDeclaration ∏
outcomeDeclaration ∏ templateDeclaration ∏
(0..1) templateProcessing ∏ (0..) stylesheet ∏
(0..1) itemBody ∏ (0..1) responseProcessing ∏ modalFeedback

Спецификация QTI зависит от двух других спецификаций IMS – Content Packaging и Meta-data. Они используются для описания метаданных вопросов и ответов и для упаковки тестов в ресурс со стандартизированной структурой. Эти спецификации определяют форматы для представления метаданных и многократно используемых обучающих пакетов (reusable packages). Поэтому онтологии этих спецификаций также должны быть включены в систему.

Было показано, что та часть онтологии QTI, которая ответственна за описание модели тестов, может быть описана в терминах OO-логики, а значит, использована в рамках системы OntoBox. Это существенно, поскольку OO-логика допускает очень эффективную реализацию, поэтому OntoBox может обрабатывать эти данные весьма продуктивно. Ранее упоминалось, что OO-логика семантически близка к системе структурных типов объектно-ориентированного языка программирования. Поэтому конкретные тесты, хранящиеся в OntoBox, могут непосредственно обрабатываться OO-языками.

Стадия B: OntoBox

На этом этапе онтологии IMS загружаются в OntoBox, специальную программную среду, поддерживающую эффективную обработку объектов в онтологиях, обеспечивающую хранение общих данных и предоставление инструментальных средств для работы с онтологиями как системы хранения данных. Воспользуемся этими возможностями для реализации системы тестирования, основанной на спецификациях IMS.

На стадии A тесты, вопросы и другие основные понятия IMS QTI были формализованы средствами OO-логики. Другими словами, они стали концептами в онтологии QTI. И конкретные тесты, и вопросы должны быть представлены в онтологии как объекты этих концептов.

OntoBox содержит решатель, написанный на Java, который очень эффективно работает с описаниями на языке OO-логики. В авторской системе тестирования OntoBox играет роль системы управления данными, в которой сохранены тесты и другие данные.

OntoBox взаимодействует с внешним миром через свой язык запросов [4]. Этот язык навигационный и имеет XPath-подобный синтаксис. Семантика языка строго связана с дескриптивными логиками: каждый запрос в нем эквивалентен некоторому концепту (формуле) SHOIN(D).

Стадия C: Методы

На данной стадии разрабатывается пакет специальных методов, поддерживающих работу со структурами данных, которые описаны в онтологии IMS. Онтологии вместе с этими методами могут составлять общую и многократно используемую библиотеку, служащую основой для разработки различных

образовательных систем, связанных со спецификациями IMS. С точки зрения концепции логических архитектур данная библиотека формирует специальный компонент логической архитектуры, в котором часть онтологии, загружаемая в OntoBox, играет роль сегмента логики.

Пакет основных методов, работающих с онтологией IMS, включает: метод, генерирующий терминологическую (без объектов) онтологию QTI в OntoBox; методы для импорта XML/QTI описания тестов и трансляции тестов в объекты онтологии QTI, хранящейся в OntoBox; для экспорта объектов из онтологии QTI, хранящихся в OntoBox, и их трансляции в тесты в формате XML/QTI; для выборки из OntoBox объектов, описывающих тесты, – для дальнейшего использования в образовательных системах, а также метод поддержки сборки тестов из базовых компонентов, который помогает составлять шаблоны тестов из основных блоков, описываемых в спецификации IMS QTI.

Взаимодействие между OntoBox и этими методами осуществляется с помощью языка запросов OntoBox QL [4] (аналогично тому, как взаимодействие с БД происходит через SQL, но OntoBox QL обеспечивает значительно более высокий, логический, уровень взаимодействия). Заметим, что в OntoBox реализована клиент-серверная технология взаимодействия, когда обмен запросами/ответами происходит поверх протокола HTTP.

Эти методы взаимодействия с онтологией QTI собраны в виде Java-библиотеки (QTI-Lib). В библиотеку включены и шаблоны типовых запросов к онтологии QTI, сформированные на языке OntoBox QL.

Стадия D: Система проверки знаний

Авторами разработана практическая систе- ма проверки знаний, основанная на OntoBox и QTI-Lib (для естественно-научного портала http://lake.baikal.ru). Эта система по существу состоит из четырех модулей: модуля для управления образовательными объектами, редактора тестов, плейера тестов, модуля авторизации и управления системой.

QTI-Lib используется для первых трех модулей. Модуль авторизации и управления основан на другой онтологии, в которой описывается общая схема организации системы безопасности и авторизации для информационных проектов. Эта онтология описывает понятия пользователя, группы, разрешения и т.д. Онтология по безопасности сопровождается пакетом модулей Security-Lib, поддерживающих работу с объектами данных типов. Заметим, что эта онтология и онтология QTI построены по одной технологии. Методы управления безопасностью и авторизацией носят универсальный характер и могут использоваться в самых разных информационных системах, включая авторскую систему проверки знаний. Общая технология использования такого пакета заключается в следующем. Через механизмы наследования к понятию пользователя системы тестирования (учитель, студент или администратор) добавляются компоненты, связанные с безопасностью и авторизацией. Наследование сразу обеспечивает возможность применения методов Security-Lib к конкретным пользователям системы тестирования. Данные, связанные с безопасностью, хранятся в отдельной онтологии авторизации, которая наследует знания из основной онтологии QTI.

Структура вопросов в QTI очень гибкая. Но блоки вопросов – это различные экземпляры нескольких абстрактных понятий, которые могут быть эффективно

обработаны в объектно-ориентированном стиле, обеспечивая возможность их повторного использования не только для онтологий, но и для методов и интерфейсов (то есть компонентов логической архитектуры), прикрепленных к этим онтологиям.

Таким образом, онтологии могут использоваться как для внутренней обработки данных в пределах информационных систем, так и для внешнего обмена данными. Это означает, что они могут заменить БД в первом случае и XML во втором. В частности, представляется весьма полезным включение в официальные тексты спецификаций привязок к онтологиям (вместо XML-привязок либо наряду с ними), что обеспечило бы более строгую семантику в спецификациях и их более точное понимание. Также можно использовать выражения OntoBox QL для портируемого описания типичных методов управления данными. И поскольку запросы в OntoBox QL на самом деле являются закодированными логическими формулами в языке дескриптивной логики SHOIN(D), получается строгая и портируемая аксиоматика для спецификаций.

Есть и другое преимущество. Базовые структуры QTI описаны как объектная модель. Предположим, что в авторской системе тестирования решено использовать БД для хранения данных о результатах теста. Тогда, очевидно, возникнет проблема отображения иерархической объектной модели QTI в плоские таблицы БД. В общем случае проблема построения такого отображения является нетривиальной. Есть много проектов (включая весьма популярные Hibernate и ADO.NET Entity Framework), в которых такие методики моделирования и отображения разработаны и реализованы. При работе в OntoBox этот шаг является ненужным, так как XML-описания тестов и вопросов непосредственно переносятся в онтологии, где могут использоваться с помощью OntoBox и, следовательно, авторской системой тестирования. Поскольку имеется огромное количество проблем, описываемых с помощью объектных моделей, технологии работы с данными, основанные на онтологиях, имеют значительные конкурентные преимущества относительно БД. С другой стороны, онтологии – намного более мощные системы, чем простые объектные БД, потому что низкий уровень объектно-ориентированной логики включен в общую среду дескриптивных логик с их мощными методами обработки знаний на терминологическом уровне. Это означает, что любой ресурс, разработанный в пределах ОО-логики, немедленно становится частью среды SW.

На основании изложенного можно сделать следующие выводы.

В данной работе исследовались возможности использования общих формальных систем, основанных на дескриптивных логиках, в качестве языков спецификаций. Работа преследует несколько целей. Во-первых, формализация спецификаций посредством дескриптивных логик обеспечивает интеграцию этих очень важных информационных ресурсов в контекст SW. Многие спецификации содержат так называемую XML-привязку – компонент, который описывает синтаксические конструкции в форме документов XML. Но XML не может описать семантику. Подъем до уровня логического формализма позволит включать в рамки спецификации не только формальное описание самой структуры, но и ее значения.

Авторы выясняли, могут ли такие формальные системы, как онтологии, в которые были преобразованы формализованные спецификации, использоваться для решения реальных задач примерно в том же стиле, как, например, БД.

Оценивалась работа OntoBox с его языком запросов как платформа для решения практических проблем. Полученный в процессе этой работы опыт оказался весьма позитивным.

Кроме того, исследовались возможности использования онтологий как хранилищ конкретных данных и объектов. Оценивалось, могут ли логически формализованные описания быть достаточно эффективными и конкурировать с такими стандартными инструментальными средствами, как БД, по крайней мере, в приложении к задачам с иерархической архитектурой и объектно-ориентированными структурами. Проект QTI очень прост в вычислительном плане, поэтому для таких оценок использовать его невозможно. Однако авторы проводили эксперименты с более сложными задачами. В частности, была построена система запросов для крупной биологической таксономии NCBI Entrez Taxonomy. Параллельно эта таксономия реализовалась в виде онтологии с запросами на языке OntoBox QL и в виде БД с запросами на языке SQL. Реализация таксономии как онтологии,

загруженной в OntoBox, показывала на группах запросов скорость в среднем в семь раз выше, чем эквивалентная реализация в СУБД MySQL.

Литература

1. Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, Peter F. Patel-Schneider (Eds.): The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press, 2003.

2. Web Ontology Language (OWL). URL: www.w3.org/ 2004/OWL (дата обращения: 24.09.2008).

3. Anton Malykh and Andrei Mantsivoda. OntoBox: Ontologies for Objects. Submitted to ISWC'09. URL: Ошибка! Недопустимый объект гиперссылки. (дата обращения: 08.09.2009).

4. Anton Malykh and Andrei Mantsivoda. A Query Language for Logic Architectures // Proceedings of International Conference on Perspectives of System Informatics'09, Novosibirsk, 2009, pp. 212–220. URL: http://meta2project.org/ru/psi09.pdf (дата обращения: 27.08.2008).

5. IMS Global Learning Consortium. URL: http://imsglobal.org/ (дата обращения: 01.11.2007).