УДК 004.62

Чистякова И.С.
Институт программных систем
НАН Украины

ИНЖЕНЕРИЯ ОНТОЛОГИЙ

В статье рассматривается понятие инженерии онтологий, её основные составляющие, такие как процесс разработки, методологии построения онтологий, понятие жизненного цикла и его структура, набор инструментов и языков для построения и поддержки онтологий. Приведена обобщенная схема жизненного цикла онтологии, на основе существующих методологий их построения, которым дан развернутый обзор.

В статті розглядається поняття інженерії онтологій, її основні складові, такі як процес розробки, методології побудови онтологій, поняття життєвого циклу і його структура, набор інструментів ті мов для побудови і підтримки онтологій. Наведена узагальнена схема життєвого циклу онтологій, що заснована на існуючих методологій та їх побудови, яким надано розгорнутий огляд.

The article discusses the concept of ontology engineering and its main components, such as the design, methodologies of creating ontology, the concept of life cycle and its structure, a set of tools and languages to build and maintain ontologies. The generalized scheme of the life cycle of ontology is based on existing methodologies for constructing them, which are given a detailed review.

***Ключевые слова**: онтология, инженерия онтологий, методологии создания онтологий.*

Введение

Современный веб был создан как веб документов. Он представляет собой информационное пространство, в котором содержится огромное количество информационных источников, содержащих взаимодополняющие, взаимозаменяющие, противоречивые и дублируемые данные. С каждым днем их количество растет в геометрической прогрессии, а релевантность - в арифметической. В результате этого возникла острая необходимость перейти от документов, к способам их описания, то есть создать структуру, которая содержит описание документа и отображает семантику его данных. Данные хранятся отдельно, и к ним существует описание, уникально их идентифицирующее. Это есть концептуальное описание предметной области, к которой будут относиться те, или иные документы.

Ни одна из существующих моделей данных не справляется с такой задачей, а в некоторых случаях даже ухудшает ситуацию, вследствие чего встал вопрос о создании новой модели данных для решения поставленной проблемы. Так возникла онтология, которая приобрела ключевую роль в области манипулирования данными с развитием информационного пространства, и количество которых стремительно растет.

Предпосылки возникновения инженерии онтологий

За последние несколько лет ко многим мировым специалистам пришло осознание следующего факта: онтологии необходимы не только в конкретной области знание-ориентированых систем. Абсолютно все популярные информационные ресурсы, программы и системы для своей корректной работы требуют наличие одной или нескольких моделей представления мира и окружающей действительности. Следовательно, в процессе жизненного цикла любой системы или программного продукта, на различных его стадиях (в особенности, на этапе разработки), становится все более и более уместно использование онтологий. К этому привело ряд следующих предпосылок.

1) Онтология может быть использована в качестве словаря (representation vocabulary). В данном случае имеется в виду не привычный словарь терминов с ярлычком «онтология», а некоторая концептуализация предметной области. Сами термины словаря представляют собой концепты (элементы ПО), а связи между ними – **семантический контекст их определений**. То есть, формулировка определения понятия является основой для существования или отсутствия связей между определяемым термином и другими терминами словаря [1]. Например, в

кулинарной книге есть Блюдо которое *состоит из* овощей и фруктов, и Картофель, который *является компонентом* блюда. При этом Семейство Паслёновые *является типом* Картофеля.

2) Онтология может быть использована как содержательная теория (content theory). Онтологическая модель имеет все средства для создания концептуального представления некоторой ПО, формально определяя её составляющие и связи внутри неё, выстраивает строгую иерархию концептов и структурирует понятия в единое общее представление. Её основная задача – определить классы, объекты и отношения, которые существуют внутри предметной области. Это дает основу для использования онтологии в качестве содержательной теории для искусственного интеллекта, позволяя применить созданные на сегодняшний день механизмы рассуждений (системы правил, нейронные сети, унификация и т.д.)

3) Онтологии дают возможность обмена знаниями (knowledge sharing). Онтологии могут применяться не только в контексте данной конкретной задачи. Знания, которые охватывает онтология могут быть использованы в различных системах, программных продуктах, приложениях, а также в смежных областях. Они могут применяться для создания новых онтологиях связанных областей, то есть несколько специфических задач могут использовать одну онтологию в качестве основной, охватывающей внутреннюю концептуальную структуру домена.

4) Онтология способна оперировать не только знаниями домена, но и реальными данными. Это означает, что в одной онтологии одновременно может существовать не только само концептуальное описание домена, но и реальные данные конкретной задачи.

Таким образом, все эти факторы обусловили появление 1) нового объекта исследования – онтологии, 2) научной дисциплины, инженерии онтологий, а также 3) ряда связанных задач (описание, поиск).

Существует несколько определений инженерии онтологий.

Инженерия онтологий в компьютерных науках – это дисциплина, которая изучает методы, методологии и средства построения и использования онтологий.

Инженерия онтологий направлена на явное извлечение и представление знаний, содержащихся в компьютерных системах предметных областей различного назначения.

Инженерия онтологий направлена на решение проблем интероперабельности, вызванных семантическими проблемами, то есть проблемами возможного неоднозначного понимания терминов той или иной предметной области.

Основываясь на работе [2] мы предлагаем следующие составляющие инженерии онтологий, дополнив его собственными пунктами:

– жизненный цикл онтологий;
– процесс разработки онтологий;
– методологии построения онтологий;
– использование онтологий;
– набор инструментов и языков для их построения и поддержки.
– Рассмотрим каждый из этих пунктов.

Жизненный цикл онтологий

Что касается жизненного цикла онтологий, то в общем он совпадает с жизненным циклом разработки любой программной систем, естественно, с учетом различных способов классификации этапов жизненного цикла и их названия. Отличия могут заключаться в видах действий, предпринимаемых на каждом из этапов.

В [2] дан исчерпывающий обзор работ, предлагающих различные методологии создания онтологий. В каждой работе описание этого процесса приводится совершенно по-разному, с различным количеством этапов и описанием действий, выполняемых на каждом из них. Опираясь на результаты этого обзора, можно сказать, что только три методологии приводят модель жизненного цикла. Это Methontology [4], UPON [6] и On-To-Knowledge [8]. Однако, проанализировав самостоятельно несколько работ, которые предлагают методологии создания онтологий без введения понятия жизненного цикла (Enterprise model approach [5], 101 method [7] и DOGMA [9]), можно сделать вывод, что все предлагаемые методологии используют одну и ту же модель, но с различной степенью детализации её стадий.

Жизненный цикл (ЖЦ) онтологий определяет последовательность стадий, которые должна пройти онтология в течении своей жизни. Согласно работе [4], главной проблемой создания знание-ориентированных систем является построение документа требований. В то же время, разработчики онтологий должны определить требования

прежде, чем создать её ядро. Таким образом, жизненный цикл онтологий гораздо ближе к классическому ЖЦ программного обеспечения, чем к ЖЦ знание-ориентированных систем.

Методология **Methontology** построена на *эволюционирующих прототипах*. Она позволяет изменять, добавлять и удалять определения в онтологии в любое время, что является наиболее приемлемым для её создания и развития. Объясняется это следующими аргументами:

1) Если применять классическую водопадную модель (Ройс, 1987), которая является традиционной в инженерии программного обеспечения, то адаптируя её к процессу создания онтологии, разработчику необходимо в самом начале определить **все** термины, а конечная версия статична, её термины нельзя модифицировать, что делает такой продукт бесполезным в рамках концепции Семантического Веба.

2) Если применять классическую инкрементную модель, то она лишь частично решает изложенные трудности, т.к. экстраполируя эту модель на процесс создания онтологии, конечному пользователю предлагается продукт, растущий «слоями», то есть все изменения вносятся в последующую версию, а текущая остается статичной, как и в адаптации классической модели.

Эволюционный прототип решает эти трудности, что обосновывает его применение авторами Methontology.

Методология **UPON** основана на адаптации *Unified Software Development Process (Unified Process)* к процессу создания онтологий. Каждый цикл **обязательно** состоит из 4 последовательно выполняющихся фаз, каждая фаза итеративна. Итераций от 1 до N, на каждой из которых формально и так же последовательно выполняется 5 рабочих процессов. Фактически же от того, на какой фазе находится цикл, зависит какие именно рабочие процессы будут реально выполняться, а какие присутствуют лишь формально. В конце каждого цикла получается новая, полноценная версия онтологии. То есть, реальный объем работ и их последовательность в цикле такая же линейная, как и в классической модели Ройса. При этом каждая стадия (фаза) повторяется несколько раз прежде, чем выполняется переход к следующей, а количество циклов определяет лишь разработчик.

Методология **On-To-Knowledge** основана на методологии C*ommonKADS*, в основе которой лежит *Knowledge Meta Process*. Его суть в систематическом внедрении знаний в готовые программы и системы.

В работе [5] предложена комплексная методология создания онтологий, которую назвали **Enterprise Model Approach**. Она создавалась под конкретную задачу, и авторы не ставили себе цель создать универсальный метод разработки онтологий. Однако, был предложен некоторый комплексный подход, в котором рассматривается последовательность из четырех шагов, которые, по мнению автора, должны быть обязательно выполнены для создания полноценной онтологии.

В результате анализа этих работ, мы решили обобщить процесс создания онтологии и привести обобщенную (возможную) схему её жизненного цикла (рис. 1). Она включает в себя основные, на наш взгляд, стадии, которые должна пройти онтология в процессе создания. Степень детализации на каждой из стадий отдается на усмотрение разработчиков. В конце каждого цикла получается готовая версия онтологии, как описано в методологии UPON [6]. При необходимости, цикл можно повторить заново.

Остановимся кратко на каждой из стадий.

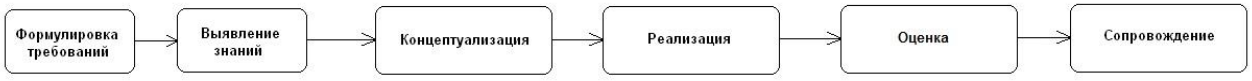

Рис.1 Обобщенная схема жизненного цикла онтологий

Формулировка требований

Какую бы из существующих методологий мы не рассматривали, в ней явным или неявным способом присутствует стадия обоснования. Некоторые авторы выносят обоснование создания онтологии за пределы жизненного цикла, некоторые включают её в первую фазу, расширяя и детализируя её. Мы же предлагаем внести её процесс разработки как полноценную стадию жизненного цикла, и часть расширенных полномочий делегировать

следующей стадии. В стадию обоснования предлагаем внести следующие действия:

1) Формулировка цели создания онтологии.
2) Выделение конечных пользователей.
3) Выделение домена и суб-доменов, которые будут описаны в онтологии.
4) Ресурсы (исполнители, программное и аппаратное обеспечение для разработки).
5) Сроки выполнения.
6) Возможность повторного использования.

В методологии UPON [6] эти пункты внесены в самый первый рабочий процесс «Требования». Формально он выполняется на каждой итерации каждой фазы. Фактически этот рабочий процесс присутствует только на первой фазе «Создание», то есть в самом начале всего цикла создания. Также в один и тот же рабочий процесс вынесены пункты, которые мы относим к следующей стадии.

В методологии On-To-Knowledge [8] на стадии «Обоснование» упоминается о выборе ресурсов, конечных пользователях и повторном использовании, но ничего не сказано о цели, сроках и самой онтологии. Выделение доменов и суб-доменов объединено с пунктами, которые мы рассмотрим в следующей стадии, и в самой методологии также рассматривается на следующей стадии цикла. Однако, отмечен такой пункт, как выделение домена для приложения, которое будет использовать онтологию. Поскольку мы не рассматриваем в нашей обобщенной схеме возможность параллельного создания и программного обеспечения для онтологии, то мы опускаем этот пункт и в обобщение он не вошел.

В методологии METHONTOLOGY [4] эта стадия вообще не входит в жизненный цикл. Она вынесена как отдельная транзакция, предфаза самого цикла. Выполняется один раз в самом начале. Описана кратко и сжато, определяются лишь цель (которая разбивается на набор задач), сроки выполнения и ресурсы. Все прочее определяется на первой стадии жизненного цикла.

В методологии Enterprise Model Approach [5] на первом шаге автор указывает о необходимости определения цели создания онтологии. Также предлагается определить не только цель, но и конечных пользователей, и рассмотреть возможность повторного использования конечной онтологии.

Мы выделили именно этих 6 пунктов в первую стадию, руководствуясь следующими причинами. Любой инженерный проект требует определения целей, сроков и ресурсов для выполнения. Даже если мы создаем новую версию, то без определения этих пунктов любой проект становится нежизнеспособным. Также нужно очертить круг конечных пользователей, т.к. именно на их нужды разработчики будут ориентироваться при формулировке требований к онтологии, равно как и предметная область, которую она будет описывать. Человеком не прекращается познание окружающего мира, что обуславливает появление новых, ранее неизвестных данных различных предметных областей. Поскольку мы имеем дело с созданием источника знаний, поэтому следует заранее предусмотреть возможность повторного использования онтологии, а также добавление новых данных, удаление старых и изменение существующих.

Выявление знаний

Согласно Груберу [11] это одна из двух обязательных стадий в процессе создания онтологии, и она должна присутствовать в любой методологии (вне зависимости от наличия или отсутствия понятия жизненного цикла), на чем мы убедились в процессе анализа существующих решений. Любая онтология должна основываться на проверенных источниках знаний, а также предусматривать повторное использование уже существующих онтологий для того, чтобы избегать дублирования информации.

Мы обобщили эту стадию до следующих пунктов:

1) Формулировка вопросов компетентности. Это контрольные вопросы, на которые итоговая онтология должна будет дать ответы.
2) Определение и анализ источников знаний (эксперты в предметной области, различные документы, литературные источники, существующие онтологии данной и смежных предметных областей).
3) Непосредственное извлечение знаний (интеллектуальный анализ текста, «мозговой штурм», интервью с экспертами и т.д.)
4) Формирование основного лексикона, а именно: формулирование фундаментальных концептов и отношений между ними, выработка непротиворечивых текстовых определений, а также идентификация терминов для таких концептов и отношений, с учетом согласования всего вышеперечисленного между собой.

Относительно пункта, который касается непосредственного извлечения знаний, следует выделить следующее. В настоящий момент существует целое научное направление, которое называется обучение онтологии (Ontology learning). В область его исследований входит создание алгоритмов, методов и средств автоматизированного извлечения знаний из текстов в естественном языке и создание онтологий из этих знаний. Как показали результаты, этот процесс является трудоемким и трудозатратным, а также требует немалого количества времени. Поэтому существует большое количество автоматизированных и полуавтоматизированых средств извлечения доменных знаний. Приведем небольшой перечень таких средств, классификация которых основана на типе входных данных [3]:

- Методы обучения онтологии из текстов. Извлечение онтологий путем применения методов анализа естественно-языковых текстов.
- Методы обучения онтологий из словарей. Принцип действия основан на использовании машиночитаемого словаря для извлечения соответствующих понятий и отношений между ними.
- Методы обучения онтологий из базы знаний. Обучение онтологии происходит на основе существующей базы знаний.
- Методы обучения онтологий из полу-структурированых данных. Онтология формируется из источников, имеющих предопределенную структуру, например XML-схемы.
- Методы обучения онтологий из реляционных схем. Онтология формируется на основе знаний, содержащихся в базах данных.

Как мы видим, в данной области существует ряд направлений исследований, в каждом из которых существуют свои наработки и результаты. Их подробный анализ представлен в публикации [3].

Что касается прочих пунктов стадии выявления знаний, Ушхольд [5] ничего не говорит о вопросах компетентности, но все остальные пункты присутствуют во втором шаге «Построение онтологии». Эта стадия выделена отдельным пунктом, тем самым автор акцентирует внимание на её значимость для всего процесса. Также автор подразумевает создание концептуальной модели предметной области на этом шаге.

Методология On-To-Knowledge также ничего не говорит о вопросах компетентности, однако предлагает завершить эту стадию созданием концептуальной модели онтологии, не ограничиваясь формированием основного лексикона.

Методология METHONTOLOGY начинает жизненный цикл с определения внутренних и внешних пользователей, что мы вынесли в первую стадию нашей обобщенной схемы, а также останавливается на формулировании вопросов компетентности. Что касается процесса извлечения знаний, то автор полагает, что данный процесс должен присутствовать постоянно, на протяжении всего жизненного цикла. А после вопросов компетентности следует сразу переходить к созданию концептуальной модели.

Извлечение знаний в методологии UPON происходит итеративно в течении первых двух рабочих процессов. Автор акцентировал внимание именно на этом процессе и структурировал его так, что по окончанию мы получаем готовую концептуальную модель онтологии, которую остается лишь реализовать в некотором формальном языке. На каждой итерации соответствующего рабочего процесса происходит извлечение небольшой порции знаний, которая анализируется, обрабатывается и добавляется в структурированный результат самого процесса (как кусочек паззла в общую картинку). Рассмотрим подробнее, как это происходит в UPON.

Есть два первых рабочих процесса «Требования» и «Анализ». Формально они выполняются на каждой из четырех фаз ЖЦ. Фактически их основная роль приходится лишь на первую фазу (иногда на первых две, то есть лишь первую половину всего цикла). Вместе совокупность их действиях составляет совокупность всех тех действий, которые мы выполняем в процессе двух первых стадий нашего обобщенного ЖЦ. По окончанию каждого рабочего процесса мы получаем некоторый результат, а именно: результатом процесса требований является сформированный *лексикон приложения*, результатом процесса анализа являются сформированные *лексикон домена*, *ссылочный лексикон* и *ссылочный глоссарий*. В таблице 1 представлены определения этих терминов.

Таблица 1
Результаты рабочего процесса ЖЦ

Лексикон – набор терминов.	$L = \{t_i\}$, где t_i - термин лексикона $i \le i_{max}$ $i_{max} = \mid L \mid$ $i \in N$
Лексикон приложения – это лексикон, у которого все термины относятся к заданному приложению APP, согласно утверждению сообщества экспертов.	$L = \{at_i \in L\} \mid pertains(at_i, APP)$, где L – лексикон, APP – приложение (наша онтология) at_i - термин приложения $i \in N$
Лексикон домена – лексикон, утвержденный сообществом экспертов.	$DL = \{dt_i \in L\} \mid pertains(dt_i, Dom)$, где Dom – домен интересов $i \in N$
Ссылочный лексикон – лексикон, утвержденный сообществом экспертов домена и приложения по согласованию с инженерами знаний.	$RL = (AL \bigcap DL) \bigcup \sigma AL \bigcup \sigma DL$ $\sigma AL = \{t_i \in AL \setminus DL \mid approved(t_i)\}$ $\sigma DL = \{t_i \in DL \setminus AL \mid approved(t_i)\}$ $i \in N$
Глоссарий – конечный набор терминов, принадлежащих лексикону, в паре с соответствующими описаниями, утвержденными рекомендованным сообществом.	$G = \{< t_i, des_i > \mid t_i \in L \wedge des_i \in DES \wedge validated(t_i, des_i)\}$, где G – глоссарий t_i - термин лексикона des_i - описание термина лексикона L – лексикон DES = {des_i} $i \in N$
Входящий глоссарий – пара термин-описание.	$g_i = < t_i, des_i >$, где g_i – входящий глоссарий $i \in N$
Ссылочный глоссарий – глоссарий, утвержденный сообществом экспертов приложения и домена по согласованию с инженерами знаний.	$RG \subseteq RL \times DES$, где $RL = \{rt_i\}$ rt_i - термин ссылочного лексикона $DES = \{des_i\}$ des_i - текстовое описание термина лексикона $i \in N$

В нашей трактовке лексикон приложения – это лексикон, который нарабатывается исключительно экспертами-участниками процесса создания онтологии. Лексикон домена – это лексикон, который уже существует в источниках данной предметной области (а также смежных областей), утвержденный сообществом экспертов. У них есть своя область пересечения, на основании которой формируется ссылочный лексикон.

Терминам, принадлежащим сформированному ссылочному лексикону, в пару подставляются соответствующие текстовые описания (определения), которые утверждены сообществом экспертов, и таким образом формируется ссылочный глоссарий. Этот глоссарий является входными данными для формирования концептуальной модели онтологии.

Схематически этот (итеративный) процесс показан на рисунке 2.

Мы приводим этот механизм потому, что считаем его одним из возможных способов выявления знаний. В разных источниках эта стадия описывается с различной степенью детализации. Некоторые авторы просто лишь декларируют, что выявление знаний обязательно присутствует в их методологии, без какого-либо описания механизма. Некоторые дают краткое пояснение, что они подразумевают под этим термином, и каким образом можно осуществить выявление знаний на практике. В методологии UPON было дано наиболее обширное пояснение этого механизма среди всех, рассмотренных нами, источников.

Концептуализация

На этой стадии выполняется всего одно действие – составление концептуальной модели предметной области.

Как уже было сказано выше, такие методологии как Enterprise Model Approach и On-To-Knowledge совмещают эту стадию со стадией выявления знаний. При этом, On-To-Knowledge предлагает внести в концептуальную модель лишь базовые понятия и отношения, а обогащать её в процессе формализации, то есть на следующей стадии.

Методология METHONTOLOGY фактически сразу переходит к её созданию, поскольку действия, которые мы описывали в предыдущих двух стадиях, в METHONTOLOGY выполняются частично предусловием всего ЖЦ, частично на самой первой стадии, за которой сразу следует построение концептуальной модели, а частично параллельно с циклом.

В методологии UPON следует остановиться на третьем из пяти рабочих процессов «проектирование». Его результатом является UML-диаграмма классов, которая играет роль концептуальной модели в UPON. В процессе категорируются концепты с помощью мета-онтологий, создается таксономия классов, свойств, концептов. Его результатом является *семантическая сеть*, которая и есть искомая концептуальная модель.

Семантическая сеть – конечный набор концептов и конечный набор взаимоотношений, установленных между концептами.

$$SN = (C, R),$$

где SN – семантическая сеть,

$C = \{c_i\}$, c_i – концепт,

$R = IsA \cup De \cup DR = \{< c_i, c_k >\} \subseteq C \times C,$

$i, k \in N$

$IsA = \{< c_i, c_k > | \; gen(c_i, c_k)\}$

$De = \{< c_i, c_k > | \; partOf\ (c_i, c_k)\}$

$DR = \{< c_i, c_k >\} | \; rel\,(c_i, c_k)\}$

gen – generalization (обобщение)

rel – association (ассоциация)

partOf – aggregation (агрегация)

Поскольку мы подробно расписали механизм извлечения знаний методологии UPON, то мы предлагаем его продолжение для составления концептуальной модели онтологии в качестве возможного варианта выполнения стадии концептуализации.

Реализация

Реализация – создание онтологии с помощью некоторого формального языка описания (OWL, OIL, RDF, SPARQL и т.д.) на основе концептуальной модели. Мы предлагаем использовать языки семейства OWL (OWL, OWL 2) как имеющие наиболее развитую семантику.

Методология METHONTOLOGY предлагает использовать фреймовые системы или дескриптивные логики. Их предложение согласуется с нашей рекомендацией по использованию языка OWL, т.к. дескриптивная логика SHOIN(D) заложена в его основу.

Методология On-To-Knowledge предлагает формализовать концептуальную модель с использованием языка OIL, при этом продолжать процесс выявления знаний (каким образом – не уточняется) для обогащения онтологии деталями и менее значимыми концептами и отношениями.

В методологии Enterprise Model Approach дается ссылка на мнение Грубера [11], гласящее, что стадии выявление знаний (в которую входит концептуализация) и реализации можно объединить. Однако сам автор считает, что любая методология, объединяющая эти две стадии должна давать серьезное обоснование такого шага. Поэтому в

Enterprise Model Approach стадия реализации вынесена отдельно и является следующим шагом после этапа выявления знаний.

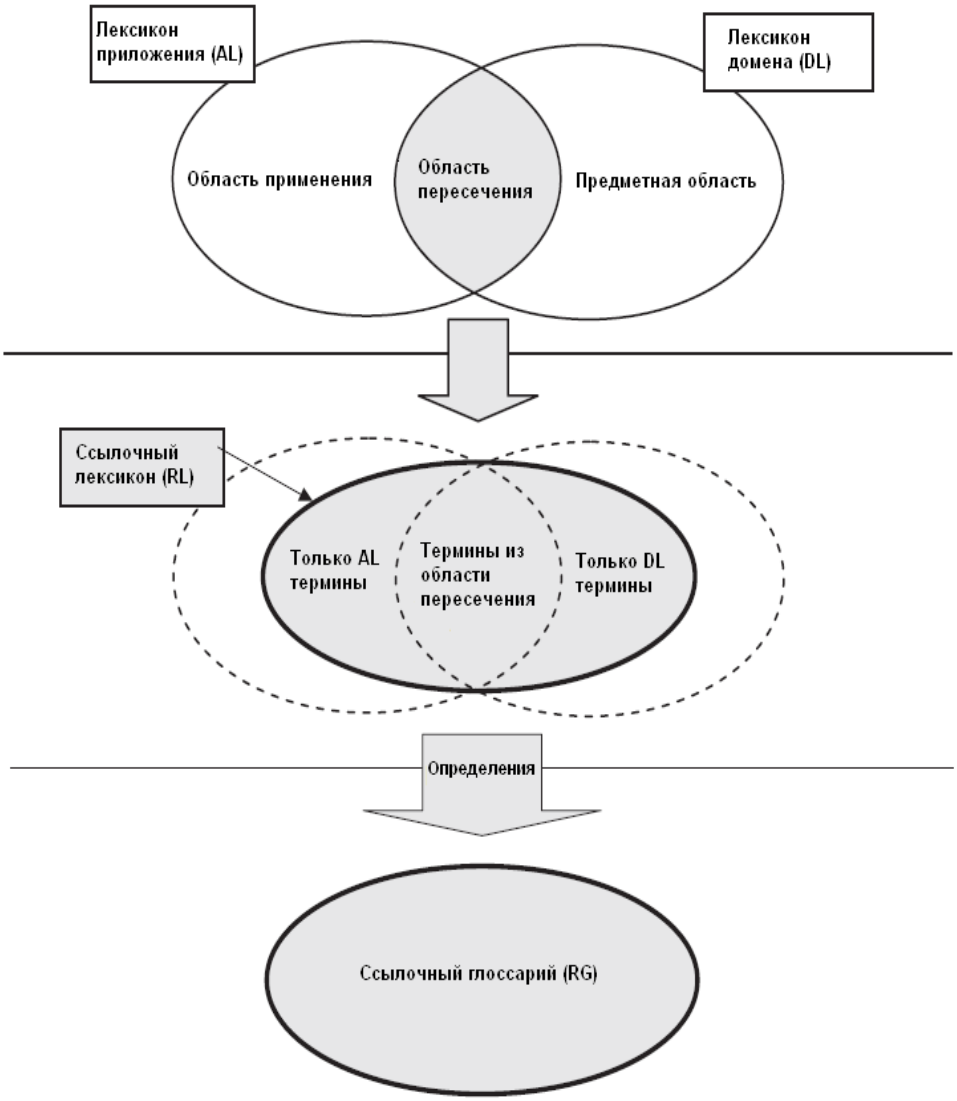

Рис. 2 Семантическая сеть

В методологии UPON четвертым (и предпоследним) является процесс реализации, который фактически выполняется во второй половине цикла. Его определение полностью совпадает с определением нашей стадии, а также рекомендуется язык OWL в качестве формального языка описания. Дается определение онтологии.

Онтология – конечный набор концептов, конечный набор взаимоотношений, установленных между концептами, конечный набор семантических аксиом.

$O = (C, R, Ax)$, где

O – онтология,

$C = \{c_i\}$,

$R = IsA \cup De \cup DR = \{< c_i, c_k >\} \subseteq C \times C$,

Ax – аксиома, $Ax = \{boolExp_j\}$, $i, k, j \in N$

Оценка

Основной задачей этой стадии является проверка соответствия созданной онтологии первоначальным требованиям (включая проверку согласно составленных вопросов компетентности). Все рассмотренные методологии единогласно утверждают, что эта стадия необходима.

Автор методологии Enterprise Model Approach ссылается на METHONTOLOGY, в которой утверждается, что оценка включает в

себя верификацию и валидацию. Под верификацией понимается проверка на корректность и правильность концептов и связей онтологии, внедрение программного обеспечения, ведение и оформление документации. Под валидацией подразумевается соответствие этих пунктов тем, которые изначально были заявлены в требованиях.

Методология On-To-Knowledge рассматривает эту стадию как такую, в которой есть возможность итерации внутри цикла, если возникает необходимость в повторной оценке.

В методологии UPON последний рабочий процесс называется «Тестирование». Он включает в себя не только оценку соответствия требованиям к онтологии, но также её работоспособность как программного продукта. Также задаются следующие критерии оценки качества:

1) Синтаксическое качество – качество онтологии, в зависимости от её формального стиля (как она написана).

2) Семантическое качество – проверка на отсутствие противоречивых концептов.

3) Прагматическое качество – насколько содержание онтологии полезно конечному пользователю, независимо от её синтаксиса и семантики.

4) Социальное качество – ссылка на онтологию извне (сколько существующих онтологий с ней связано).

Сопровождение

Аналогично сопровождению любого программного обеспечения. В любой произвольный промежуток времени любой конечный пользователь может обратиться к созданной онтологии. В результате своих действий он должен получить достоверные и актуальные данные. Основным действием этой стадии является полноценное и своевременное изменение онтологии в соответствии с изменениями реального мира. Онтология должна эволюционировать, для чего процессы обновления-удаления-вставки требуют строгих правил, что является основой процесса сопровождения. Как отдельная стадия из рассматриваемых методологий присутствует только в On-To-Knowledge и METHONTOLOGY. Однако в последней лишь сказано о том, что такая стадия присутствует после реализации, без каких-либо дальнейших инструкций. В UPON процессом сопровождения является новый виток жизненного цикла и выпуск обновленной версии текущей онтологии. Мы считаем, что эти два процесса никак не связаны друг с другом и не являются взаимоисключающими. Наш обобщенный ЖЦ начинает выполняться с самого начала при создании новой версии, однако постоянно сопровождать старую версию можно параллельно с разработкой новой.

Автор Enterprise Model Approach не упоминает ничего о сопровождении, декларируя свою методологию как каркасную, которую можно взять за основу при разработке собственной. В ней присутствует лишь то, без чего не может обойтись ни одна методология, а все остальное остается на индивидуальное усмотрение. Вместе с авторами METHONTOLOGY уделяют много внимания документированию, однако ограничиваются утверждением лишь о том, что все концепты и отношения должны быть хорошо описаны[5], [10].

Согласно [4] не существует полноценных методологий документирования процесса создания онтологий, а также самих концептов и отношений. METHONTOLOGY декларируется как первая и единственная, чей процесс включает составление полноценной документации всего жизненного цикла, включая документацию по онтологии.

Процесс разработки онтологий

В работах, посвященных методологиям создания онтологий, неоднократно встречались определения процесса разработки онтологий. Как правило, все они сводились к тому, что процесс их создания состоит из некоторых выполняемых шагов, в результате которых получается полноценная, готовая к использованию онтология. Например, авторы

METHONTOLOGY определяют его как набор некоторых транзакций, которые необходимо выполнить для построения онтологии [4]. При этом, процесс не включает в себя порядок выполнения этих транзакций, он определяет лишь перечень – не более.

Многие методологии, в основе которых отсутствует понятие жизненного цикла, процесс создания онтологии определяют как последовательность действий, которые составляют их суть. Например, Ontology development 101 описывает предпосылки и цели создания онтологии, её определение, список компонентов, а дальше – последовательность шагов, прямых практический инструкций и фундаментальных правил построения онтологии «с нуля» [7]. В методологии отсутствует жизненный цикл, но есть процесс создания. Аналогично Methodology for the Design and Evaluation of Ontologies [10] предлагает набор мотивационных сценариев. Каждый из них представляет собой набор проблем, адресуемых существующим онтологиям, а также набор интуитивно возможных решений этих проблем. Разрешение этих проблем составят информационное содержание новой онтологии (или будет добавлено в существующую). Процесс разрешения проблем мотивационных сценариев является процессом создания онтологий.

Мы определяем **процесс разработки (создания) онтологий** как выявление и четкое определение всех компонентов онтологии с последующим их упорядочением согласно концептуальной модели данных и формализацией её с помощью заранее выбранного языка описания, а также построение экстенсионалов.

Процесс создания онтологий включает в себя следующие пункты:

1) **Четкая спецификация ПО и требований к создаваемой онтологии.** Прежде, чем приступить к созданию онтологии, следует четко определить предметную область, ее масштаб, степень ее детализации, для чего, кого и как будет использоваться создаваемая онтология. Должна ли удовлетворять онтология требованиям повторного использования?

2) **Построение глоссария терминов** (терминов понятий, свойств, связей и пр.) с предоставлением их определений на естественном языке и указанием их синонимов, акронимов и других характеристик. В начале важно получить полный список терминов, не беспокоясь о пересечении понятий, которые они представляют, об отношениях между терминами, о возможных свойствах понятий или о том, чем являются понятия – классами, свойствами или отношениями. Определение терминов предоставляет прочную основу для однозначного понимания моделируемой предметной области всеми лицами, принимающими участи в создании и использовании онтологии.

3) Выявление и четкое и достаточно глубокое определение **понятий**. На этом этапе следует четко выяснить, что описывает это понятие: сущность, свойство или связь. Если это понятие-сущность (класс), то обладает ли оно интенсионалом и экстенсионалом или только одним из них. Например, понятие-сущность «язык» в зависимости от контекста может иметь только экстенсионал (например, в библиотечных информационных системах). В этом случае понятие «язык» целесообразно представить в виде управляемого словаря. В других ситуациях «язык» имеет интенсионал и экстенсионал (например, в лингвистических системах). Естественно, что понятие-сущность невозможно полностью определить без указания множества свойств, которые их характеризуют, а также множества взаимосвязей между ними. Наконец, следует иметь в виду, что любое понятие в зависимости от предметной области может быть сущностью, свойством сущности или связью между сущностями. Всегда задавайте себе вопрос: «Что это – сущность, свойство или связь?

4) Выявление и четкое определение множества **свойств** (атрибутов,

характеристик), характеризующих каждую сущность. Обязательное выделение множества обязательных/необязательных свойств и свойств, идентифицирующих сущности. Следует специфицировать область допустимых **значений** свойств и других ограничений на значения свойств. Другими словами, свойства также могут иметь свойства (обязательное, идентифицирующее, область допустимых значений, тип, мощность).

5) Выявление и четкое определение **родовидовых зависимостей** (связей) между понятиями и, тем самым построение таксономий на множестве понятий. Определите такие свойства таксономических связей, как непересекаемость, декомпозиция и полнота декомпозиции. Одна и та же сущность может участвовать во многих таксономиях. Более того, возможны ситуации, когда в одной таксономии присутствует множественное наследование.

6) Выявление и четкое определение других **произвольных бинарных связей** между понятиями с указанием к какому типу связей они относятся (эквивалентность, часть/целое, агрегация, ассоциация, причина-следствие и т.д.). Необходимо указать **мощность** окончаний связей (1:1, 1: M, M:N) и их **факультативность** (обязательная, факультативная). Бинарные связи также могут обладать свойствами рефлективности, транзитивности и симметричности). Тем самым строится тезаурус. Имейте в виду, что сами связи также могут иметь таксономическую структуру (например, связь между человеком и организацией «являться сотрудником» является обобщением связи «являться старшим научным сотрудником».

7) Выявление и четкое определение **аксиом** (правил, прикладных ограничений), характеризующих углубленную семантику понятий, атрибутов и связей. Тем самым строится описание онтологии.

8) **Построение экстенсионалов (экземпляров)**. Последний шаг – это создание одного (или даже несколько) экземпляров онтологии. Это подразумевает создание экземпляров сущностей (классов) с указанием значений всех свойств этой сущности, а также всех необходимых связей между ними.

Это далеко не полный перечень того, что включается в себя процесс создания онтологий. Мы сочли за необходимое остановиться именно на этих пунктах, поскольку они в общих словах описывают выявление знаний, которые впоследствии лягут в основу онтологии.

Приведенное нами описание процесса разработки во многом согласуется с теми описаниями, которые мы встречали в литературе. Основное различие состоит в том, что подходы к описанию процесса достаточно сильно отличаются друг от друга, но смысл действий и конечный результат не меняются. Каждая из методологий акцентирует внимание на каком-то конкретном действии, весьма скромно описывая все остальные. Мы же дали собственное, независимое от контекста, задач и предметной области, определение процесса разработки онтологии, обобщили и описали максимальное количество действий, которые входят в его состав, оставляя на откуп создателей онтологий наличие и последовательность выполнения тех или иных действий в процессе разработки.

4. Методологии построения онтологий

Современная наука предлагает множество методов и методологий построения онтологий. Некоторые из них уже упоминались в данной публикации. Однако, это далеко не полный перечень методик, которые доступны широкому кругу разработчиков онтологий. В публикации [2] приведен исчерпывающий анализ методологий, доступных к использованию на сегодняшний день. В работе дана краткая характеристика каждой из исследованных публикаций, сформулирован перечень критериев оценки методологий, а также сравнительная таблица, основанная на данном перечне, которую мы и приводим для ознакомления (табл. 2).

Таблица 2

Методологий построения онтологий и их характеристики

Методологии	**Тип разработки**	**Совместное конструирование**	**Поддержка повторного использования**	**Степень зависимости от приложений**	**Рекомендации по жизненному циклу**	**Стратегии определения концептов**	**Детали методологии**	**Поддержка совместимости**
TOVE	На основе этапов	-	-	полу-независимое	-	средняя	некоторые	-
Enterprise model approach	На основе этапов	-	-	независимое	-	средняя	некоторые	-
METHONTOLOGY	Прототип развития	-	-	независимое	+	средняя	достаточно	-
KBSI IDEF5	Прототип развития	-	-	независимое	-	не ясно	некоторые	-
Ontolingua	Модульное развитие	+	-	независимое	-	не ясно	некоторые	+
Common KADS and KACTUS	Модульное развитие	-	-	зависимое	-	сверху-вниз	недостаточно	-
PLINIUS	Директивы	-	-	независимое	-	снизу –вверх	некоторые	-
ONIONS	Модульное развитие/ Директивы	-	-	зависимое	-	не ясно	недостаточно	+
Mikrokosmos	Директивы	-	-	зависимое	-	основана на правилах	некоторые	-
MENELAS	Директивы	-	-	зависимое	-	CG	недостаточно	-
SENSUS	Не отмечено	+	+	полунезависимое	-	снизу-вверх	некоторые	+
Cyc methodology	Прототип развития	-	+	независимое	-	не ясно	некоторые	-
UPON	Прототип развития	-	+	независимое	+	средняя	некоторые	-
101 method	Прототип развития	-	+	независимое	-	согласие разработчика	некоторые	-
On-To-Knowledge	Прототип развития	-	-	зависимое	+	средняя	некоторые	-

В приведенной публикации пояснения к таблице не приводятся, однако, совершенно очевидно, что понятия, приводимые в таблице, пришли в инженерию онтологий из инженерии программного обеспечения.

Тип разработки онтологии дает нам краткое понимание процесса создания онтологии. В него включено:

- прототип развития (присутствует ЖЦ или же его неявное присутствие);
- тип развития на основе этапов (процесс разработки происходит в несколько этапов и на этом завершается)
- модульное развитие (схоже с предыдущим пунктом, только вместо этапов создаются отдельные модули, которые потом включаются в онтологию)
- директивы (процесс создания описывается простым перечнем действий, которые необходимо выполнить для создания онтологий; такой перечень может достигать более 100 пунктов)

Степень зависимости от приложения показывает, насколько онтология независима от той системы, для которой она была создана (возможность ее независимого использования в других приложениях и системах). Сюда включено:

- зависима (результирующая онтология полностью зависима от системы или приложения, вне которого использовать ее нельзя);
- полунезависима (результирующая онтология частично зависима от системы или приложения, вследствие чего некоторые ее компоненты могут быть повторно использованы);
- независима (результирующую онтологию можно совершенно независимо использовать в различных приложениях и системах).

Детали методологии показывают, насколько подробно и понятно (с точки зрения автора таблицы) для использования описана методология. Выделено два пункта: описаны лишь некоторые детали или же дано достаточно подробное описание.

Использование онтологий

Использование онтологий лишь косвенным образом относится к инженерии онтологий. Мы посчитали необходимым выделить это отдельным пунктом, основываясь на том факте, что онтология может выступать в качестве повторно используемого компонента (как целиком, так и её части).

Существуют различные аспекты использования онтологий: хранение, создание реестров, поиск, повторное использование и т.д. Мы здесь кратко осветим только один из них - оперирование онтологиями. При этом мы кратко раскроем суть основных операций над онтологиями, взяв при этом за основу статью [27].

Если для решения той или иной задачи необходима онтология, которой нет, но она может быть получена из других существующих, то в этом случае возникает необходимость выполнения последовательности операций над существующими онтологиями с целью построения необходимой. Ниже мы кратко обсудим некоторые из существующих операций над онтологиями. Следует отметить, что терминология относительно операций еще не устоялась, некоторые различные операции могут иметь одинаковое название, и наоборот, одна и та же операция может пониматься по-разному. Не говоря уже о том, что перевод на русский язык англоязычных названий операций также еще не устоялся. В связи с последним обстоятельством мы для всех названий операций будем приводить их англоязычные варианты.

Сопоставление онтологий (ontology matching) является перспективным направлением в решении проблем семантической неоднородности. Сопоставление призвано решать задачи нахождения соответствия между семантически связанными сущностями двух онтологий. Такое сопоставление может использоваться для решения различных задач, например, слияние онтологий (ontology merging), получение ответов на запросы, трансляция данных, навигация по семантическому вебу. Таким образом, сопоставление онтологий позволяет осуществлять оперирование данными и знаниями, представленными в сопоставляемых онтологиях. Учитывая важность этого аспекта в манипулировании онтологиями, был создан сайт (http://www.ontologymatching.org/), содержащий литературу, презентации, оценки, проекты, полезные ссылки в этой области.

Слияние онтологий (ontology merging). Это создание новой согласованной онтологии, которая объединяет в себе две (а может быть и больше) других. Многие исследователи считают, что слияниям могут подвергаться онтологии из одной тематической предметной области**.** Общепринятым требованием также является то, что новая онтология содержит все знания из исходных онтологий с наименьшими возможными изменениями. Однако это требование в некоторых случаях не может быть полностью выполнено, так как исходные онтологии могут быть несовместимыми. В этом случае для достижения целостности (совместимости) результирующей онтологии следует пожертвовать требованию полноты. Объединенная онтология может содержать новые понятия и отношения, которые служат в качестве «связующего звена» между терминами оригинальных онтологий.

Отображение онтологии (ontology mapping). Отображение одной онтологии в другую – это

функция преобразования одной онтологии в другую (способ перевода объектов одной онтологии в другую), либо сам результат такого преобразования. Часто это означает отображение понятий и отношений. В простейшем случае понятия и отношения отображаются в соответствующие понятия и отношения другой онтологии. В более сложных случаях примитивные (атомарные) понятия и отношения могут отображаться в сложные понятия и отношения и наоборот. Отображение может быть частичным в том смысле, что не все понятия исходной онтологии отображаются в результирующую. В частности, это означает, что в исходной онтологии существует подонтология, для которой существует полное отображение. Отображение не должно порождать никаких несоответствий.

Согласование онтологий (alignment) - это процесс отображения онтологий в обоих направлениях. Согласование, как и отображение, может быть лишь частичным. Спецификация согласования называется артикуляцией (articulation).Согласование заключается в том, чтобы установить различные виды соответствий между двумя онтологиями, а затем сохранить исходные онтологии вместе с информацией о найденных соответствиях с тем, чтобы в дальнейшем использовать информацию о взаимосвязях онтологий. В некоторых случаях допускается добавление новых понятий и отношений к исходным онтологиям с тем, чтобы найти подходящее двустороннее отображение. Также предлагаются варианты, когда используется степень (точность) устанавливаемых соответствий.

Уточнение онтологии (ontology refinement). Под уточнением онтологии понимается такое сопоставление онтологии A с другой онтологией B, что каждому понятию из онтологии A ставится в соответствие эквивалентное ему понятие в B. Примитивным понятиям из онтологии A могут соответствовать непримитивные понятия онтологии B.

Унификация онтологии (ontology unification). Онтология приводится к некоему каноническому (эталонному) представлению. Для унификации должна задаваться исходная онтология, которая приводится к результирующей согласно заданной канонической онтологии. Задача унификации множества исходных онтологий становится актуальной при работе с гетерогенными онтологиями

Интеграция онтологий (ontology integration). Это операция нахождения одинаковых частей (то есть установление соответствия) двух разных онтологий, A и B, при разработке новой онтологии C. Интеграция позволяет выполнить перевод между онтологиями A и B, и, таким образом, позволяет осуществлять взаимодействие между двумя системами, где одна использует онтологию A, а другая - онтологию B. Новая онтология C может заменить онтологии A и B или может использоваться в качестве промежуточной онтологии для перевода между двумя онтологиями. В зависимости от поставленных целей уровень интеграция может меняться от согласования и до унификации. Отличие интеграции от слияния – merging заключается в следующем:

- интеграция устанавливает соответствие, а слияние дает одну результирующую онтологию;
- интеграция, как правило, действует над онтологиями различных доменов, а слияние – одного домена.

Наследование онтологии (ontology inheritance). Наследование означает, что онтология A наследует все из онтологии B. Она наследует все понятия, отношения, ограничения, аксиомы, и дополнительные знания, содержащиеся в онтологии, не внося при этом какой-либо несогласованности. Если так можно выразиться, это родовидовое отношение между онтологиями, когда вид наследует все свойства рода и, при этом, имеет свои дополнительные видовые свойства. Родовая онтология описывает общие знания, а видовая описывает специализированные знания, базирующиеся на общих.

Набор инструментов и языков для построения и поддержки онтологий

Как следует из вышеизложенного, построение онтологий – очень сложный и трудоемкий процесс. Для его облегчения начали создаваться специальные инструменты для автоматизации процесса создания онтологий. Самые первые из них обладали обширными возможностями, которые включали в себя не только концептуализацию и реализацию, но и возможность документирования онтологии, а также проверки непротиворечивости её составных частей. За последние годы количество и качество инструментария возросло на несколько порядков.

На данный момент существует следующий перечень общих возможностей редакторов:

- пользовательский интерфейс представлен одним из двух (или обоими) интерфейсами:
- web-приложение
- локальное приложение
- графический интерфейс для редактирования и навигации
- поддержка редактора формальных аксиом и сложных выражений
- коллективная разработка

Это лишь обобщенный перечень, который присущ всем существующим редакторам. Для дополнительных сведений следует рассматривать каждый из них индивидуально.

В работе [12] представлен подробный обзор существующих программных сред построения онтологий. В нем рассмотрено три группы инструментов, а также небольшой сравнительный анализ внутри каждой из них. Мы приведем лишь перечень и краткую характеристику групп:

1) Инструменты построения онтологий.

Обеспечивают процесс создания «с нуля». Согласно [12], помимо общих функций редактирования и просмотра выполняют поддержку документирования онтологий, импорт и экспорт онтологий разных форматов и языков, поддержку графического редактирования, управление библиотеками онтологий и т.д.

К ним относятся: Ontolingua [13], Protégé [14], OntoEdit [15], OilEd [16], WebOnto [17] и т.д. Краткую характеристику этих и других инструментов можно найти в упомянутой работе [12].

2) Инструменты для отображения, выравнивания и объединения онтологий.

Согласно [12], инструменты объединения онтологий помогают пользователям найти сходство и различие между исходными онтологиями и создают результирующую онтологию, которая содержит элементы исходных онтологий. Для достижения этой цели они автоматически определяют соответствия между концептами в исходных онтологиях или обеспечивают среду, где пользователь может легко найти и определить эти соответствия. Эти инструменты известны как инструменты отображения, выравнивания и объединения.

Автор [12] делит их на подгруппы по следующим признакам:

- для объединения двух онтологий с целью создания одной новой (PROMPT [18], Chimaera [19], OntoMerge [20]);
- для определения функции преобразования из одной онтологии в другую (OntoMorph [21]);
- для определения отображения между концептами в двух онтологиях, находя пары соответствующих концептов (например, OBSERVER [22], FCA-Merge [23]);
- для определения правил отображения для связи только релевантных частей исходных онтологий (ONION [24]).

3) Инструменты аннотирования на основе онтологий.

Согласно [12], важнейшим предусловием реализации целей семантического Web является возможность аннотировать Web-ресурсы семантической информацией. В связи с этим в последние годы инструменты инженерии онтологий эволюционируют в сторону разработки инструментов аннотирования на основе онтологий.

К ним относятся: MnM [25], SHOE Knowledge Annotator [26] и т.д.

За последние годы этот перечень пополнился не одним инструментом. Однако, приведенные выше программные средства остаются основополагающими и наиболее полными на сегодняшний день инструментами разработки онтологий.

Выводы

Инженерия онтологий – новое и перспективное научное направление, развитие которого обусловлено стремительным ростом современного информационного пространства. Ключевую роль в этой области играет онтология.

В работе приведены основные составляющие инженерии онтологий, каждой из которых дана характеристика. Приведен сравнительный анализ методологий, среди которых более подробно рассмотрены те, в основе которых лежит жизненный цикл. В результате этого анализа было выявлено, что ни одна из существующих методологий не является полностью развитой. Внимание акцентировано лишь на наиболее важной (с точки зрения самой методологии) части, в то время как остальные описаны недостаточно подробно. В работе сделана попытка унифицировать полученные сведения, в результате чего приведена обобщенная модель жизненного цикла онтологий.

Выполнен обзор существующих методов создания онтологий, а также программного инструментария процесса создания онтологий.

На данный момент инженерия онтологий является молодым, стремительно развивающимся научным направлением, в котором ведутся активные научные исследования по разработке новых и улучшению старых теоретических и практических средств создания, редактирования, поддержки актуальности сведений онтологий.

Список использованных источников

1. Chandrasekaran, B., Josephson, J.R., Benjamins, V.R. What are ontologies, and why do we need them? // IEEE Intelligent Systems. –1999. – №14 (1). – P. 20 – 26.

2. Rizwan Iqbal, Azrifah Azmi Murad, Aida Mustapha and Nurfadhlina Mohd Sharef. An Analysis of Ontology Engineering Methodologies: A Literature Review // Research Journal of Applied Sciences, Engineering and Technology. –2013. – № 6(16). – P. 48 – 62.

3. Asunción Gómez-Pérez, David Manzano-Macho. A survey of ontology learning methods and techniques [Электронный ресурс]. – Режим доступа: http://www.sti-innsbruck.at/sites/default/files/D1.5_0.pdf

4. Fernandez M., Gomez-Perez A., Juristo A. Methontology: From Ontological Art Towards Ontological Engineering // Spring Symposium on Ontological Engineering (AAAI): Technical Report SS-97-06.

5. Uschold M., King M. Towards a Methodology for Building Ontologies // Proceeding of the Workshop on Basic Ontological Issues in Knowledge Sharing. – 1995.

6. Nicola A.D., M. Missikoff and R. Navigli. A proposal for a unified process for ontology building: UPON Database and Expert Systems Applications.

7. Noy N. Ontology development 101: A guide to creating your first ontology / Noy N., McGuinness // Technical Report. – 2001.

8. Sure Y., Staab S., Studer R. On-To-Knowledge Methodology / Staab S., Studer R // Handbook on Ontologies. – 2003. – № 6. – P. 135 – 152.

9. Jarrar M. Ontology Engineering -The DOGMA Approach / Jarrar M., Meersman R. // In Advances in Web Semantics I. – 2008. – Chapter 3.

10. Gruninger M. Methodology for the design and evaluation of ontologies / Gruninger M., Fox M.S. // Workshop on Basic Ontological Issues in Knowledge Sharing. – 1995.

11. Gruber T. Towards principles for the design of ontologies used for knowledge sharing / Gruber T. // International Journal of Human-Computer Studies. – 1995. – №43 (5/6) – P. 907 –928.

12. Овдей О.М. Обзор инструментов инженерии онтологий / Овдей О.М., Проскудина Г.Ю. // [Электронный ресурс]: – Режим доступа: http://www.elbib.ru/index.phtml?page=elbib/rus/journal/2004/part4.

13. Инструмент построения онтологий Ontolingua [Электронный ресурс]. – Режим доступа: http://www-ksl.stanford.edu/

14. Инструмент построения онтологий Protégé [Электронный ресурс]. – Режим доступа: http://protege.stanford.edu/

15. Инструмент построения онтологий OntoEdit [Электронный ресурс]. – Режим доступа: http://link.springer.com/chapter/10.1007%2F3-540-48005-6_18.

16. Инструмент построения онтологий OilEd [Электронный ресурс]. – Режим доступа: http://oiled.semanticweb.org/building.

17. Инструмент построения онтологий WebOnto [Электронный ресурс]. – Режим доступа: http://kmi.open.ac.uk/projects/webonto.

18. Инструмент построения онтологий PROMPT [Электронный ресурс]. – Режим доступа: http://protege.cim3.net/cgi-bin/wiki.pl?Prompt.

19. Инструмент построения онтологий Chimaera [Электронный ресурс]. – Режим доступа: http://www.ksl.stanford.edu/software/chimaera.

20. Инструмент построения онтологий OntoMerge [Электронный ресурс]. – Режим доступа: http://cs-www.cs.yale.edu/homes/dvm/daml/ontology-translation.html.

21. Инструмент построения онтологий OntoMorph [Электронный ресурс]. – Режим доступа: http://www.isi.edu/~hans/ontomorph/presentation/ontomorph.html.

22. Инструмент построения онтологий OBSERVER [Электронный ресурс]. – Режим доступа: http://sid.cps.unizar.es/OBSERVER.

23. Инструмент построения онтологий ONION [Электронный ресурс]. – Режим доступа: http://ontolog.cim3.net/cgi-bin/wiki.pl?ONION.

24. Stumme G. FCA-Merge: Bottom-up merging of ontologies / Stumme G., Medche A. // 7th Int. Conf. on Artificial Intelligence, (IJCAI'01), Seattle, WA. – 2001. – P.225 – 230.

25 Vargas-Vera M., Motta E., Domingue J., Lanzoni M., Stutt A., Ciravegna F. MnM: Ontology-Driven Tool for Semantic Markup // European Conf. on Artificial Intelligence (ECAI 2002). In Proc. of the Workshop Semantic Authoring, Annotation & Knowledge Markup (SAAKM 2002). – Lyon France. – July, 22–23, 2002.

26. Heflin J. A Portrait of the Semantic Web in Action/ Heflin J., Hendler J.// IEEE Intelligent Systems, March/April. – 2001. – P. 54 – 59.

27. Operations on Ontologies [Электронный ресурс]. – Режим доступа: http://www.obitko.com/tutorials/ontologies-semantic-web/operations-on-ontologies.html.

Сведения об авторе:

Чистякова Инна Сергеевна – аспирантка Института программных систем НАН Украины. Научные интересы: Semantic Web, семантическая интеграция данных, онтологические модели данных.

E-mail: inna_islyamova@ukr.net