



О ПОДХОДЕ К СОЗДАНИЮ ОНТОЛОГИЙ НА ОСНОВЕ СИСТЕМНО-ОБЪЕКТНЫХ МОДЕЛЕЙ ПРЕДМЕТНОЙ ОБЛАСТИ

А.А. СЛОБОДЮК
С.И. МАТОРИН
С.Н. ЧЕТВЕРИКОВ

*Белгородский
государственный
национальный
исследовательский
университет*

*e-mail:
anna-slobodyuk@yandex.ru
matorin@bsu.edu.ru,
Chetverikov_SN@belsbyt.ru.*

Современный этап развития онтологического инжиниринга предполагает развитие существующих и создание новых, более эффективных методологий и подходов к построению онтологий. В статье приведен обзор основных и наиболее распространенных методологий создания онтологий, проанализированы их проблемы и недостатки, а также рассмотрен вариант их классификации. Предложен новый подход к созданию онтологий на основе УФО-моделей предметной области. Также в статье рассмотрен порядок записи УФО-модели с помощью языка RDF.

Ключевые слова: онтология, онтологический инжиниринг, классификация методологий, METHONTOLOGY, DILIGENT, UPON, жизненный цикл онтологии, подход «Узел-Функция-Объект», RDF.

Развитие онтологического инжиниринга сопровождается появлением ряда нетривиальных проблем и исследовательских задач. В частности, для создания наиболее эффективных приложений с целью решения прикладных задач требуется использование онтологий соответствующих предметных областей. Существующие на сегодняшний день готовые онтологии зачастую оказываются малоприспособленными для построения на их основе информационных систем, решающих прикладные задачи, что обуславливает потребность в быстрой и эффективной разработке новых онтологий. Однако, процесс создания онтологий предметных областей сопровождается рядом сложностей, в числе которых:

- высокие требования к участникам разработки (экспертам предметной области и инженерам по знаниям);
- длительность и трудоемкость процедур;
- сложность формализации и, как следствие, автоматизации процессов разработки;
- вероятная субъективность результативной онтологии.

Поэтому одной из актуальных задач онтологического инжиниринга является разработка эффективных методов и подходов к построению онтологий предметных областей. В данной области был проведен ряд базовых исследований, затрагивающих такие вопросы, как выделение основных этапов создания онтологии, анализ их особенностей, построение алгоритмов прохождения указанных этапов, а также разработка конкретных техник выполнения каждого этапа.

Каждое из подобных исследований обладает уникальными особенностями и разной степенью детализации, однако в литературе встречаются попытки охарактеризовать известные методологии построения онтологий с точки зрения определенных критериев – например, соответствия методологии разработки программного обеспечения, определенным стандартам (например, IEEE 1074-1995) или методологиям инженерии знаний.

Однако в существующих исследованиях, несмотря на попытки систематизации информации о методологиях построения онтологий, не разрабатывается и не используется какая-либо однозначная классификация таких методологий; тогда как подобный способ характеристики и сравнения существующих методологий позволил бы:

- наглядно представить сходства и различия методологий;
- обозначить преимущества и недостатки каждого подхода;
- выявить некоторые общие тенденции развития методологий;
- облегчить выбор оптимальной методологии создания онтологии в конкретной ситуации.

Поэтому на основе существующих работ в области методологий построения онтологий и их сравнительных описаний нами разработан вариант классификации, основанный на наиболее существенных, на наш взгляд, признаках. В соответствии с данной клас-



сификацией все существующие методологии разработки онтологий можно условно разделить на представленные ниже группы.

1. По типу разработки онтологии:
 - методологии централизованной разработки;
 - методологии децентрализованной разработки.
2. По степени зависимости от приложения:
 - зависимые;
 - полузависимые;
 - независимые.
3. По типу используемой модели жизненного цикла (ЖЦ) онтологии:
 - без указания модели ЖЦ онтологии;
 - с итеративной моделью ЖЦ онтологии;
 - с моделью ЖЦ онтологии на основе эволюционного прототипирования.
 - с совпадающей с моделью ЖЦ приложения.
4. По возможности формализации:
 - предусматривающие способы формализации;
 - не предусматривающие формализации.
5. По типу исходных данных, запускающих процесс создания онтологии:
 - на основе вопросов компетенций;
 - на основе описания сценариев (вариантов использования);
 - на основе модели предметной области;
 - на основе исходной онтологии.
6. По стратегии выделения концептов предметной области:
 - снизу-вверх (bottom-up);
 - сверху-вниз (top-down);
 - от середины (middle-out);
 - сочетающие различные стратегии.

На наш взгляд, основными, наиболее сформировавшимися на сегодняшний день и ставшими базовыми в области создания онтологий предметных областей методологиями можно считать: скелетную методологию Ушолда и Кинга, методологию, разработанную Грюнингером и Фоксом, METHONTOLOGY, On-To-Knowledge, KACTUS, DILIGENT, методологию на основе подхода SENSUS, а также UPON. Их сравнительная характеристика в соответствии с приведенной классификацией приведена в таблице 1.

Рассмотрим детально характеристику перечисленных в таблице 1 методологий и подходов к созданию онтологий. В 1995 году Ушолд и Кинг разработали методологию построения онтологий, которая в дальнейшем получила название «скелетной» [1]. Данная методология предполагает четыре этапа – определение цели, непосредственно разработка онтологии, оценка и документирование. Недостатком скелетной методологии является то, что она не подразумевает развитие онтологии в соответствии с каким-либо жизненным циклом. В дальнейшем это приводит к проблематичности расширения, модификации онтологии, ее сравнения и слияния с другими онтологиями. Также в данной методологии не указаны возможные техники выполнения отдельных этапов построения онтологии. Еще одним недостатком методологии Ушолда и Кинга является отсутствие указаний на способы формализации знаний, полученных в ходе разработки онтологии.

Известной работой в области исследования процесса создания онтологий считается методология, предложенная Грюнингером и Фоксом [2]. В ее основе лежит идея использования сценариев (вариантов использования), инициирующих разработку онтологии. Также на начальном этапе разработки онтологии, согласно Грюнингеру и Фоксу, применяется формулировка так называемых неформальных вопросов компетенции, ответы на которые должна давать разрабатываемая онтология. Последующие этапы разработки предполагают выявление основных терминов онтологии, перевод вопросов компетенции в формальное описание, определение формальных спецификаций определений терминов и аксиом, а также выделение условий

Таблица 1

Сравнительная характеристика методологий построения онтологий

Методология	Тип разработки	Степень зависимости от приложения	Используемая модель жизненного цикла онтологии	Возможность формализации	Исходная информация для построения онтологии (модель предметной области, вопросы компетенции, задачи)	Стратегия выделения концептов
Uschold and King	Централизованный	Независимая	Не упоминается	Не упоминается	Вопросы компетенции	От середины (middle-out)
TOVE (Grunniger and Fox)	Централизованный	Полузависимая	Не упоминается	Логика	Сценарии, вопросы компетенции	От середины (middle-out)
SENSUS	Централизованный	Полузависимая	Не упоминается	Семантические сети	Исходная онтология	Снизу-вверх (bottom-up)
METHONTOLOGY	Централизованный	Независимая	Эволюционное прототипирование	Фреймы и дескрипционная логика	Сценарии, задачи	От середины (middle-out)
KACTUS	Централизованный	Зависимая	Совпадает с моделью жизненного цикла приложения	Не упоминается	Исходная база знаний	Снизу-вверх (bottom-up)
ОТК	Централизованный	Полузависимая	Итерационная	Не упоминается	Вопросы компетенции	От середины (middle-out). Возможна поддержка стратегии снизу-вверх (bottom-up)
UPON	Централизованный	Полузависимая	Итерационная	Семантические сети	Модель предметной области	От середины (middle-out)
Faceted Approach	Централизованный	Полузависимая	Не упоминается	Не упоминается	Модель предметной области	Снизу-вверх (bottom-up) и сверху-вниз (top-down)
DILIGENT	Децентрализованный	Зависимая	Эволюционное прототипирование	Не упоминается	Исходная онтология	В зависимости от решаемой проблемы



полноты ответов на сформулированные вопросы, получаемых с помощью онтологии. Характерной особенностью данной методологии является ее полузависимый от приложения характер, поскольку результатная онтология ограничивается в применении сценариями, определенными приложением. Также авторы не рассматривают каких-либо конкретных техник, применяемых в рамках методологии – например, техники получения вопросов компетенций.

Еще одной распространенной методологией построения онтологий является METHONTOLOGY [3]. Эта методология разрабатывалась с учетом положений стандарта IEEE 1074-1995 и достаточно детально описывает все этапы создания онтологий, условно разбитые на три группы – действия по управлению проектом, действия, ориентированные на разработку и техническую поддержку. METHONTOLOGY характеризуется независимостью от конкретного приложения и предполагает развитие онтологии по пути развития прототипов. Данная методология считается одним из наиболее зрелых подходов и рекомендуется ассоциацией FIPA. Однако METHONTOLOGY, так же, как и вышеупомянутые методологии, не содержит детального описания подпроцессов создания онтологии.

В рамках проекта EU IST-1999-10132 On-To-Knowledge была разработана одноименная методология создания онтологий. Основным предназначением методологии On-To-Knowledge (ОТКМ) является построение систем управления знаниями, основанных на онтологиях и в частности – организация процессов, управляющих разработкой и начальной настройкой (установкой) онтологий [4]. Базисом методологии ОТКМ является представление всего мета-процесса получения знаний в виде пяти основных шагов, каждый из которых также делится на определенные этапы: технико-экономическое обоснование, начало (стартовая часть), доработка (усовершенствование), оценка, применение и развитие. Основной особенностью, и, на наш взгляд, недостатком методологии On-To-Knowledge является ее специализация в области разработки онтологий для централизованных систем. Также, поскольку ОТКМ предполагает цикличность этапов доработки, оценки, применения и развития, становится затруднительно спрогнозировать сроки окончания разработки онтологии, а также оценить возможность применения результата того или иного цикла для решения поставленных задач.

Среди множества разработок в области создания онтологий выделяется методология, основанная на подходе SENSUS. Ее отличительная характеристика состоит в том, что новая онтология строится путем добавления концептов и последующего сокращения уже существующей, заранее заданной онтологии, а именно SENSUS – онтологии, предназначенной для использования в операциях на естественном языке. Последовательность шагов, выполняемых для получения новой онтологии на основе онтологии SENSUS, приводится в работе [5]. Так, предполагается выделение множества концептов-«семян», их привязка к онтологии SENSUS и последующее включение других терминов (причем фактически применяется восходящий способ выделения концептов предметной области – от наиболее специфичных к наиболее общим). В заключение разработчиками принимается решение о включении в онтологию вложенных деревьев некоторых узлов-концептов. Методология, основанная на подходе SENSUS, на наш взгляд, также обладает определенными недостатками. Среди них можно выделить высокие требования к создателям онтологии, значительную степень субъективного влияния участников разработки на результат процесса создания онтологии, а также отсутствие наработок по конкретным действиям на каждом этапе построения онтологии.

Оригинальный подход, предполагающий тесную взаимосвязь между разработкой онтологий и приложений, был предложен в рамках проекта The ESPRIT CACTUS. Авторы предлагают строить онтологии на основе («извлекать из») баз знаний конкретных приложений с помощью метода «снизу-вверх», то есть путем абстракции концептов. Недостатками такого подхода является то, что в основных исследованиях по этой теме [6] не приводится каких-либо конкретных способов формализации получаемых и обрабатываемых знаний. В сочетании с фактом отсутствия четких и подробных алгоритмов выполнения отдельных этапов построения онтологии, это может затруднять применение методологии CACTUS на практике.



В середине 2000-х годов был разработан подход к построению онтологий, получивший название DILIGENT (Distributed, Loosely-controlled and evolving Engineering of Ontologies) [7]. Данная методология подразумевает пять основных этапов (шагов) разработки онтологии: построение, локальная адаптация, анализ, ревизия и локальное обновление. Характерной чертой DILIGENT является отсутствие требований к полноте исходной онтологии-основы. Основным процесс создания новой онтологии заключается в том, что пользователи локально модифицируют исходную онтологию, а группа контроля осуществляет мониторинг локальных изменений, оценивает их значимость, и, при необходимости, реализует соответствующие изменения в ядре основной онтологии. Несмотря на широкое распространение, которое получила данная методология, она обладает значительным недостатком – ее применение требует привлечения существенного числа специалистов. При этом во всех действиях преобладает эвристическая составляющая, а возможность применения автоматизированных средств или формальных подходов практически отсутствует. Еще одним недостатком методологии DILIGENT может считаться вероятность того, что исходная онтология недостаточно охватывает предметную область, и, как следствие, не может быть применена по назначению без доработок. При этом сроки реализации участниками процесса таких доработок не поддаются точной оценке.

В отдельную группу исследований, можно объединить методы построения онтологий с помощью машинного обучения, и, в частности, использование методов лингвистического анализа текстов для последующего автоматического построения онтологий. Однако, согласно мнению Е.А. Ермакова [8], одной из основных причин отсутствия современных «...прикладных программ, использующих «онтологии в сильном смысле» является несовершенство применяемых алгоритмов извлечения знаний из текстов и зачастую сомнительное качество самих источников информации. Это влечет за собой необходимость последующей «ручной» верификации и редактирования полученных прототипов онтологии экспертом. Также при использовании методов лингвистического анализа текстов возникает проблема обеспечения соблюдения принципа расширяемости, сформулированного Грубером. Предполагается, что для разрабатываемой онтологии должен быть определен «механизм расширения (ограничения) совместно используемых словарей понятий без нарушения целостности системы» [9]. Исследований, посвященных решению данной проблемы в методологиях, использующих методы лингвистического анализа текстов, на сегодняшний день практически нет.

Некоторой попыткой решения описанной проблемы можно считать фасетный подход, разработанный Рубен Прието-Диаз. Данный подход предлагает сочетать нисходящую и восходящую методики проектирования онтологии, а также использовать методы анализа предметной области. Однако результат разработки в этом случае не включает в себя формальные определения концептов и аксиом. Поэтому требуются более тщательные исследования аспекта преобразования моделей предметной области в формализованные онтологии, что отмечается и самими авторами [10].

В работе [11] описан подход к процессу создания онтологии, включающий в себя построение на начальном этапе некоторой абстрактной модели предметной области, в качестве которой могут выступать use-case-диаграммы на языке UML. Данная методология носит название UPON и разработана в соответствии с унифицированным процессом разработки программного обеспечения (Software Development Unified Process, UP), что позволяет сопоставлять разработку онтологии с разработкой программного обеспечения. В частности, методология UPON подразумевает иерархическое представление процесса создания онтологии – в виде циклов, которые состоят из четырех фаз, каждая из которых подразделяется на итерации, а те, в свою очередь – на работы. Процесс создания онтологии здесь носит итеративный и инкрементальный характер, что, несмотря на ряд преимуществ, таких как, например, гибкость и масштабируемость результативной онтологии, существенно увеличивает нагрузку на всех участников процесса разработки. Для поддержки рассмотрения отдельных положений проекта разработки онтологии в методологии UPON предлагается использовать язык UML. Однако, несмотря на то, что UML-диаграммы в данном случае позволяют достичь наглядности и простоты представления целей создания онтологии, весь процесс разработки оказывается нацеленным на решение



лишь ряда конкретных задач. Это ведет к затруднению обеспечения соблюдения одного из ключевых положений онтологического инжиниринга – возможности использования одной и той же онтологии для решения широкого круга задач, в том числе и заранее неопределенных.

Таким образом, каждый из рассмотренных подходов к построению онтологий обладает теми или иными недостатками. Это обуславливает необходимость разработки новых, более эффективных методологий и подходов, которые позволили бы:

- обеспечить наибольшую объективность и полноту разрабатываемой онтологии;
- снизить долю эвристических алгоритмов и «ручных» операций при разработке онтологии;
- формализовать методы и разработать на их основе автоматизированные средства создания онтологий;
- сохранить возможность использования получаемой онтологии для использования различными приложениями, в том числе для решения заранее не определенных задач.

Одним из подходов, решающих указанные задачи, на наш взгляд, может стать использование в качестве источника сведений о предметной области, необходимых для создания онтологии, системно-объектных моделей в терминах «Узел-Функция-Объект» (УФО-моделей), получаемых в результате применения системного подхода с тем же названием (т.е. УФО-подхода). В частности, предлагается извлекать некоторые факты о предметной области из УФО-моделей и представлять их с помощью специализированного языка описания онтологий. Рассмотрим данный подход подробнее.

В основе УФО-подхода и базирующегося на нем УФО-анализа лежит представление той или иной системы в виде трехэлементной конструкции «Узел-Функция-Объект». Под узлом понимают систему в виде структурного элемента, «перекрестка» всех ее связей. Функция представляет систему как функциональный элемент, играющий заданную роль для поддержания надсистемы с помощью балансирования данного узла. Объект представляет систему в виде конкретного образования, субстанциального элемента, реализующего функцию. Комбинации из подобных УФО-элементов представляют собой системно-объектные модели (УФО-модели) [12].

Важной особенностью подхода, предполагающего использование УФО-моделей в качестве источника фактов для построения онтологии, является возможность разработки методов и средств автоматизации процесса построения онтологий предметных областей. Для этого рассмотрим связь УФО-подхода и онтологического инжиниринга с точки зрения формализации.

Исходным синтаксисом обмена, базисом для множества специализированных языков представления онтологий является RDF (Resource Description Framework) – универсальный язык для представления знаний в Сети. Он позволяет представлять знания в самом общем виде, при этом являясь «универсальным средством для обмена данными между разными программами» [13]. В основе RDF лежит представление фактов в качестве триплетов вида «Субъект–Отношение(предикат)-Объект». Среди множества вариантов записи (представления) RDF-фактов наиболее удобной для восприятия человеком является Нотация 3 (N3), поскольку имеет достаточно компактный формат записи [14]. Далее предлагается рассматривать связь УФО-подхода именно с RDF.

В литературе встречается несколько способов формализации УФО-подхода [15, 16, 17]. В основе подхода, описанного в работах [15, 16], лежит сопоставление образующей в теории паттернов с УФО-элементом в УФО-подходе. Указанное сопоставление графически можно изобразить следующим образом (рис. 1).



Рис. 1. Графические формализмы – образующая и УФО-элемент

- Для описания УФО-моделей с помощью RDF необходимо рассмотреть три аспекта:
- Наличие связей между УФО-элементами в УФО-модели;
 - Способность УФО-элемента иметь связи определенного типа;
 - Наличие у УФО-элемента конкретного объекта и функции.



Рассмотрим конфигурацию, состоящую из двух образующих и одного соединения между ними. Представим графически также два УФО-элемента, соединенные некоторой связью, по аналогии с описанной конфигурацией. И сравним полученную конструкцию с RDF-триплетом (рис. 2).

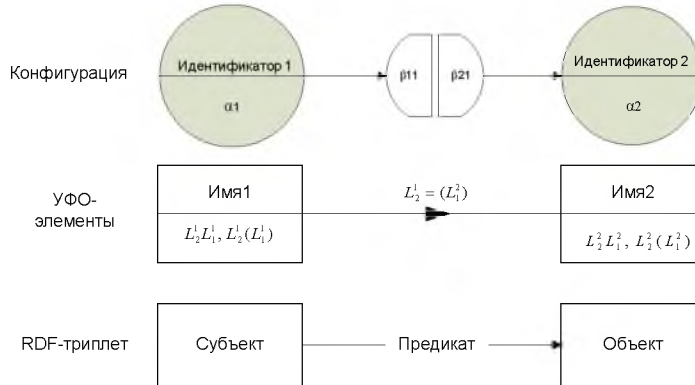


Рис. 2. Соответствие конфигурации, УФО-элементов и RDF-триплета

Данные конструкции имеют некоторые общие черты, что позволяет сопоставлять их друг другу. Так, в RDF-триплете субъект и объект представляются в виде имен конкретных или абстрактных сущностей. Аналогичным признаком является идентификатор у образующей. Каждый УФО-элемент имеет имя, которое позволяет однозначно определять его в пределах модели. При представлении УФО-элемента с использованием терминов теории паттернов имя элемента может служить идентификатором соответствующей образующей. Отсюда следует, что имена УФО-элементов могут быть использованы в качестве субъектов и объектов в RDF-триплетах.

Важной составляющей RDF-триплета является предикат – отношение между двумя сущностями. Такое отношение является направленным. Это позволяет сопоставить предикату в RDF-триплете соединение, входящее в структуру конфигурации в терминах теории паттернов. При этом, как видно из рисунка 2, аналогичным элементом будет являться и связь между двумя УФО-элементами.

С использованием положений, разработанных в работе [18], представим УФО-элементы в виде следующих конструкций:

$$\begin{aligned}
 e_1 &= \langle (L_1?, L_1!), (S_1, S_1^0, L_{1\tau}), (n_1, \alpha_1, \beta_1!) \rangle, \\
 e_2 &= \langle (L_2?, L_2!), (S_2, S_2^0, L_{2\tau}), (n_2, \alpha_2, \beta_2?) \rangle.
 \end{aligned}
 \tag{1}$$

Преобразование связи между двумя элементами в RDF-триплет должно иметь вид некоторой функции:

$$F(e_1, e_2) = \begin{cases} \langle n_1 : "Pass" + \beta_1! + "To" : n_2 \rangle, \\ \langle n_2 : "Get" + \beta_2? + "By" : n_1 \rangle, \\ \beta_1! = \beta_2?. \end{cases}
 \tag{2}$$

Соответственно, в общем виде для двух УФО-элементов, обладающих общими связями, указанная функция принимает вид:

$$F_1(e_i, e_j) = \begin{cases} \bigcup_{k=1}^K \langle n_i : "Pass" + \beta_{ik}! + "To" : n_j \rangle, \\ \bigcup_{k=1}^K \langle n_j : "Get" + \beta_{ik}! + "By" : n_i \rangle, \\ \forall k = 1..K, \exists p > 0, \beta_{jp}? = \beta_{ik}!. \end{cases}
 \tag{3}$$

где K – число связей между УФО-элементами e_i и e_j .

Еще одной возможностью извлечения из УФО-модели RDF-фактов является анализ наличия у узла УФО-элемента определенных связей. Так, следует учитывать, что связь у образующей имеет некоторый показатель связи. Аналогично связь между УФО-элементами, которая может рассматриваться как входящая связь узла одного элемента, так и как исходящая узла другого элемента, имеет тип из классификации связей. Такой показатель (тип связи) можно также рассматривать в качестве имени некой абстрактной сущности. Следовательно, факт наличия связи у узла УФО-элемента тоже можно представить в виде RDF-триплета (рис. 3).

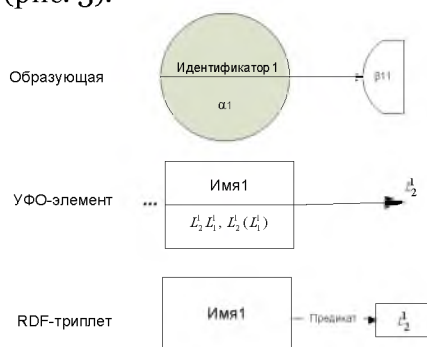


Рис. 3. Соответствие образующей, УФО-элемента и RDF-триплета

В формальном виде с учетом принятых ранее обозначений это можно представить с помощью некоторой функции:

$$F_2(e) = \begin{cases} \bigcup_{q=1}^Q \langle n : CanGet : L_q? \rangle, \\ \bigcup_{y=1}^Y \langle n : CanPass : L_y! \rangle. \end{cases} \quad (4)$$

Где Q – число элементов множества L? (количество входных связей узла), а Y – число элементов множества L! (количество исходящих связей узла).

При переводе УФО-модели в RDF-факты следует учитывать наличие у УФО-элемента конкретного объекта. Объект УФО-элемента обладает рядом свойств α . Данные факты пригодны для записи на RDF. При этом субъект будет представлен именем $n \in N$ из множества имен объектов; предикат соответственно может быть обозначен выражением «имеет свойство»; объектом будет являться конкретное свойство из множества α :

$$F_3(e) = \bigcup_{l=1}^L \langle n : hasProperty : \alpha_l \rangle \quad (5)$$

где L – число элементов множества α .

Неотъемлемой частью УФО-элемента является функция. В качестве субъекта RDF-триплета в этом случае может выступать имя объекта УФО-элемента, в качестве объекта – имя (нотация) самой функции. Формально это представляется функцией вида:

$$F_4(e) = \langle n : Executes : S^0 \rangle \quad (6)$$

Таким образом, некоторая УФО-модель предметной области может быть представлена с помощью RDF. Для этого к УФО-модели необходимо применить функцию, общий вид которой представлен в формуле:

$$F(M_{УФО}) = \{ \forall e_i, e_j \in M_{УФО}, \bigcup_{i=1}^3 F_i(e_i) \cup F_4(e_i, e_j) \} \quad (7)$$

Описанный способ формализации записи УФО-модели на языке RDF является первым этапом для построения онтологии предметной области по УФО-модели. В настоящее время ведется работа над развитием предложенного подхода.

Список литературы



1. <http://www.aii.ed.ac.uk/~entprise/enterprise/ontology.html>.
2. Gruninger M., Fox M.S. Methodology for the Design and Evaluation of Ontologies // Workshop on Basic Ontological Issues in Knowledge Sharing. – Montreal, 1995.
3. Gomez-Perez A., Fernandez-Lopez M., de Vicente A. Towards a Method to Conceptualize Domain Ontologies // ECAI96 Workshop on Ontological Engineering. – Budapest, 1996. pp.41–51.
4. Загоруйко Ю.А. Методы и методологии разработки, сопровождения и реинжиниринга онтологий // Труды Симпозиума «Онтологическое моделирование» под ред. Л.А. Калиниченко. – Звенигород, 2008. С.167–200.
5. Swartout B., Ramesh P., Knight K., Russ T. Toward Distributed Use of Large-Scale Ontologies // Symposium on Ontological Engineering of AAAI. – Stanford (California), Mars, 1997.
6. Bernaras A., Laresgoiti I., Corera J. Building and reusing ontologies for electrical network applications // Proc. European Conference on Artificial Intelligence (ECAI_96). – Budapest, Hungary, 1996. pp. 298–302.
7. Pinto H. Sofia, Staab S., Tempich Chr. DILIGENT: Towards a fine-grained methodology for Distributed, Loosely-controlled and evolving Engineering of oNTologies // In Proceedings of the 16th European Conference on Artificial Intelligence (ECAI). – 2004, № 7
8. Ермаков, А.Е. Автоматизация онтологического инжиниринга в системах извлечения знаний из текста [Текст] / А.Е. Ермаков // Диалог: материалы ежегод. Междунар. конф., Бекасово, 4-8 июня 2008 г. – М.: РГГУ, 2008. – Вып.7 (14): Компьютерная лингвистика и интеллектуальные технологии. – С.154-159.
9. Палагин А.В., Петренко Н.Г. Системно-онтологический анализ предметной области // УСиМ. – 2009. – № 4. – С.3-14.
10. <https://users.cs.jmu.edu/prietorx/Public/publications/BulidOntologiesRPD-ER2002.doc>.
11. De Nicola A., Missikoff M., Navigli R. A Proposal for a Unified Process for Ontology Building: UPON // Lecture Notes in Computer Science. – 2005. – Volume 3588. – pp.655-664.
12. Маторин С.И., Попов А.С., Маторин В.С. Моделирование организационных систем в свете нового подхода «Узел-Функция-Объект» // НТИ. Сер.2. – 2005. – №1. – С. 1-8.
13. Segaran T., Evans C., Taylor J. Programming the Semantic Web. – O'Reilly Media, 2009. 302 с.
14. Berners-Lee T., Connolly D., Kagal L., Hendler J., Schraf Y. N3Logic: A Logical Framework for the World Wide Web // Journal of Theory and Practice of Logic Programming (TPLP). – Special Issue on Logic Programming and the Web, 2008
15. Ельчанинов Д.Б., Маторин С.И. О формализации системологических понятий средствами теории паттернов // Искусственный интеллект. – 2002. – №2. – с. 116-124.
16. Жихарев А.Г., Маторин С.И. Метод формализации организационных знаний // Искусственный интеллект и принятие решений. – 2011. – № 2. – С.12-18.
17. Михелев М.В., Маторин С.И. Формализация UFO-элементов с помощью алгебраического аппарата пи-исчисления // Научные ведомости БелГУ. Сер. Информатика. – 2010. – №19(90). – Выпуск №16/1. – С.145-149.
18. Зимовец, О.А. Интеграция средств формализации графоаналитических моделей «Узел-Функция-Объект» / О.А. Зимовец, С.И. Маторин; НИУ БелГУ // Искусственный интеллект и принятие решений. – 2012. – №1. – С. 95-102.

ABOUT APPROACH FOR BUILDING ONTOLOGIES BASED ON UFO DOMAIN MODELS

A.A. SLOBODYUK
S.I. MATORIN
S.N. CHETVERIKOV

*Belgorod National
 Research University*

*e-mail:
 anna-slobodyuk@yandex.ru
 matorin@bsu.edu.ru
 Chetverikov_SN@belsbyt.ru*

Today evolution of the ontological engineering involves the development of existing methodologies and approaches to building ontologies and creation some new, more effective ones. This article provides an overview of the most common methodologies for creating ontologies, as well as analysis of their problems and disadvantages. A variant of the classification of methodologies and approaches for building ontologies is described. Also in this article a new approach for building ontologies based on UFO domain models is proposed and the procedure of represent UFO domain models by RDF is described.

Key words: ontology, ontological engineering, classification of the methodologies, METHONTOLOGY, DILIGENT, UPON, ontology life cycle, UFO approach, RDF.