МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
«САМАРСКИЙ ГОСУДАРСТВЕННЫЙ АЭРОКОСМИЧЕСКИЙ УНИВЕРСИТЕТ
ИМЕНИ АКАДЕМИКА С.П. КОРОЛЕВА
(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)»

Антология онтологии

Электронная подборка научных статей

С А М А Р А

2010

УДК 004.9(075)
ББК 32.97

Составитель: **Боргест Николай Михайлович**

Подборка научных статей из отечественных и зарубежных печатных и электронных журналов подготовлена для магистерской программы «Проектирование, конструкция и CALS-технологии в авиационной технике» по направлению 160100.68 «Авиастроение». Данная антология дополняет содержание дисциплины «Онтология проектирования».

Подборка статей осуществлена на кафедре конструкции и проектирования летательных аппаратов СГАУ.

СОДЕРЖАНИЕ

ПРЕДИСЛОВИЕ

Образовательный контент дисциплины «Онтология проектирования» для магистерской программы "Проектирование, конструкция и CALS-технологии в авиационной технике" по направлению 160100.68 "Авиастроение" включает в себя подборку научных статей.

Как известно, **антология** (греч. anthologia, от ánthos — цветок и légo — собираю) - сборник, содержащий избранные произведения (изречении, стихи, статьи) или их фрагменты, написанные различными авторами. А **онтология** (*ontologia* от др.-греч. ὄντος — сущее, то, что существует и λόγος — учение, наука) , в транскрипции изучаемой дисциплины, - это не столько философская наука о сущем, о бытие, сколько спецификация концептуализации. Онтология в данном контексте есть ничто иное, как попытка всеобъемлющей и детальной формализации некоторой области знаний с помощью концептуальной схемы. Обычно такая схема состоит из структуры данных, содержащей все релевантные классы объектов, их связи и правила (теоремы, ограничения), принятые в этой области.

Из названия данной брошюры видно, насколько важно не только понимание самих терминов, их сути и смысла, но и точности, соблюдения корректности принятых соглашений, в том числе, в их написании (грамотности). Ведь в русском языке звучание этих двух отличающихся по смыслу терминов одинаково. В написание же разница лишь в первой букве. Поэтому в реальной жизни, а онтология изучает именно то, что реально существует, многое имеет значение: и размер, и буква, и запятая (если к тому же вспомнить известную фразу: «Казнить нельзя помиловать»).

Собранные тексты научных статей погружают в результаты конкретных научных исследований в данной предметной области, показывают глубину решаемых проблем и, безусловно, полезны в понимании полноты и границ изучаемого научного направления. Сборник состоит из опубликованных статей отечественных и зарубежных авторов в печатных и электронных журналах, трудах различных конференций и симпозиумов.

Прочтение приведенных статей необходимо совмещать с изучением учебного пособия «Онтология проектирования» и выполнением лабораторных работ по методическим указаниям «Решение проектных задач с помощью онтологических систем», которые подготовлены на кафедре конструкции и проектирования летательных аппаратов СГАУ (национальный исследовательский университет).

Некоторые проблемы широкого внедрения онтологий в IT и направления их решений

Е.М. Бениаминов
125993, ГСП-3, Москва, Миусская площадь, д. 6, РГГУ, ebeniamin@yandex.ru, http://beniaminov.rsuh.ru

Труды Симпозиума "ОНТОЛОГИЧЕСКОЕ МОДЕЛИРОВАНИЕ". М.:ИПИ РАН, 2008, с.71-82.

В статье рассматривается современное состояние развития и применения онтологий в информационных технологиях. Особое внимание уделяется проблемам, которые затрудняли широкое внедрение онтологий, внедрение и использование библиотек онтологий. Выделяются направления, которые могут стимулировать массовое внедрение онтотехнологий. К ним относятся применение технологий Web 2.0 при построении библиотек онтологий, использование стандартов Semantic Web и применение открытых языков представлений онтологий, формируемых самими пользователями.

1. Ведение

Системы, основанные на знаниях, – это довольно широкая область в компьютерной науке, в которой складываются собственные методы и принципы и которая оказывает существенное влияние на развитие информационных технологий. Один из существенных принципов, сложившихся в этой области – это разделение декларативных (непроцедурных) и императивных (процедурных) знаний и создание баз декларативных знаний. Тенденция такого разделения в программировании привела к принципам объектно-ориентированного программирования и логического программирования. В базах данных декларативные знания выделяются в виде описании схем баз данных. Особое место базы декларативных знаний получают в связи с развитием Интернет. В 1991 году вводится (Gruber T. R., [1]) термин «онтология» для обозначения связного фрагмента декларативного знания и использования его в информационных технологиях. До этого этот термин использовался в философии.

Много лет назад я занимался алгебраическими моделями баз данных, и мне стало понятно, что схемы баз данных являются элементами особых структур, которые позже были названы онтологиями, и нужны специализированные системы, поддерживающие процессы формирования и отладки многомодульных библиотек онтологий. Я стал заниматься приложением математической теории категорий к моделированию онтологий и разработкой принципов построения системы формирования и отладки онтологий [2]. Десять лет назад Леонид Андреевич Калиниченко любезно указал мне на систему Ontolingua – первую систему в Web для работы с онтологиями. С тех пор я с большим интересом слежу за этой темой [3].

Целью этой статьи является охарактеризовать состояние и развитие систем онтологий в Веб, начиная с 1995 года, постараться определить некоторые причины трудностей внедрения и использования таких систем и определить некоторые направления развития и преодоления трудностей.

2. Общая характеристика онтологий

Определение онтологии было дано Т. Грубером [1]. Здесь я не буду его повторять, а выделю лишь некоторые свойства онтологий, существенные для дальнейшего изложения.

• Онтологии представляют собой спецификации на формальном языке, в которых фиксируются договоренности группы специалистов о том, что как называется в их области и каким свойствам (соотношениям) удовлетворяет.

• На логическом уровне каждой онтологии соответствует некоторая теория (сигнатура+аксиомы), а иногда и некоторая фиксированная модель (множества+операции+отношения). Вопросы к онтологии интерпретируются как запросы к соответствующей ей теории (модели).

• Онтологии, как правило, строятся по модульному принципу: при определении новой онтологии могут использоваться уже ранее построенные онтологии.

• Онтологии должны быть удобны для понимания специалистами и интерпретироваться системами при использовании.

Простейшими, но распространенными и очень важными примерами онтологий, являются системы классификаций. Всем известно значение классификационных систем К. Линнея в биологии и Д. Менделеева в химии. Более того, имеется представление, что в любой области знаний имеется своя классификационная система, которая лежит в ее основе. Естественно, эти классификационные системы довольно устойчивы, но все-таки развиваются по мере развития наук.

Мне посчастливилось быть участником междисциплинарной конференции «Первой Всесоюзной школы-семинара по методологии и теории классификации» 25-31 октября 1979 г. в поселке Борок, где были широко обсуждены проблемы и принципы классификаций в различных науках. Вопросы, поднятые на этой конференции, и энтузиазм ее участников ярко описаны в [4].

Сейчас в области информационных технологий, связанных с онтологиями, либо мало говорят о принципах классифицирования (хотя в некоторых работах, связанных с системой Cyc [5], [6] есть, например, упоминание о машинном обучении), оставляя эти вопросы специалистам в предметных областях, либо навязывают конкретные системы классификаций. При этом разрабатываются графические редакторы, делающий процесс построения систем классификаций более наглядным, и разрабатываются стандарты машиночитаемых документов по классификациям. Теперь, все больше, классификационные системы используются компьютерными программами при решении различных задач.

Большой интерес представляют более сложные онтологии, которые разработчики системы Cyc [5], [6] называют микротеориями. В общем случае, в онтологии задаются имена классов, имена свойств, типы значений свойств, некоторые экземпляры классов, функции (операции) и отношения между классами и элементами, а также аксиомы, связывающие элементы онтологий. Сложные онтологии строятся по модульному принципу и разрабатываются коллективами специалистов в предметных областях. Поэтому в системах, поддерживающих разработку сложных онтологий должна в какой-то степени поддерживаться многоверсионность, тестирование и отладка онтологий. Естественно, при этом возникают библиотеки (базы) онтологий, и необходимы специальные средства для работы с этими библиотеками.

В онтологии представляется фрагмент взгляда на некоторые контексты миров, представляющие интерес для специалистов в данной предметной области. Естественно, что некоторые онтологии из одной библиотеки онтологий могут быть несовместимыми между собой, так как соответствуют разным ситуациям (объединение всех онтологий библиотеки не образует непротиворечивую теорию). Функция библиотеки онтологий – предоставить удобные модули и среду для формирования онтологий конкретных задач или приложений.

Примерами сложных онтологий являются онтологии, описывающие схемы баз данных. Технология работы со сложными онтологиями может использоваться при

формировании сложных схем баз данных, согласовании совместной работы нескольких баз данных, при создании распределенной базы данных.

Процесс тестирования и отладки онтологий предполагает, что в системах формирования онтологий должен обеспечиваться некоторый удобный для понимания специалистом способ представления данных об онтологии или следствий введенных в онтологии предположений. В общем случае, это может быть либо язык запросов, либо набор конкретных запросов, ответы на которые программно формируются и представляются пользователям. При этом следует иметь в виду, что теории, соответствующие онтологиям, как правило, неполны, то есть имеются утверждения, истинность или ложность которых не следует из аксиом, введенных в онтологии.

3. Основные примеры серверов онтологий и систем, использующих онтологии в Веб

3.1. Система Cyc

Cyc — это закрытый проект по созданию объёмной онтологической базы знаний, позволяющей программам решать сложные задачи из области искусственного интеллекта. Автор - Дуглас Ленат [7]. Начало разработки - 1984 г. На текущий момент база знаний Cyc содержит 2,2 миллиона утверждений (фактов и правил), описывающих более 250 тысяч термов, включая почти 15000 предикатов [8]. Модули представлены в виде микротеорий. Имеется открытый фрагмент онтологии Cyc (OpenCyc [9]) и его представление в Web [10]. Более подробный обзор этой системы приведен в [11], в котором отражены история проекта Cyc, использование системы, Cyc и концепция Web 3.0, язык CycL и критика системы.

3.2. Система Ontolingua

Web-сервер Ontolingua для хранения онтологий и межмашинного обмена онтологиями разработан в 1995 г. лабораторией KSL Стэнфордского университета. Имеется большая библиотека онтологий в открытом доступе для произвольных пользователей на станице [12]. Интересные демонстрационные примеры применения системы указаны на странице: [13]. В этих примерах показывается, как строятся онтологии задач на основании библиотек онтологий из различных областей знаний.

Об этой системе я подробнее писал в статье [3], поэтому здесь упомяну лишь о существенных изменениях, произошедших за последнее время.

Одна из функций системы Ontolingua – это интеграция с другими системами представления знаний (в частности, с Cyc). Эта функция выполняется подсистемой OKBC (Open Knowledge Base Connectivity) на базе языка межмашинного обмена знаниями KIF (Knowledge Interchange Format). В последнее время в качестве стандарта языка обмена онтологиями в Ontolingua используется и язык OWL, предложенный в проекте Semantic Web.

3.3. Проект Semantic Web (Web 3.0)

Проекту Semantic Web посвящен доклад В.Ф. Хорошевского на данном симпозиуме, поэтому здесь этот проект подробно не обсуждается. Начало проекта относится к 2001 году. Следует отметить, что работы над этим проектом активизировались в последнее время, возможно, в связи с поддержкой DARPA. В рамках этого проекта в развитие принципов языка XML для представления онтологий были разработаны языки RDF, OWL, язык запросов SPARQL и язык правил SWRL. Эти языки становятся стандартами для межмашинного обмена онтологиями и работы с онтологиями.

3.4. The World FactBook

The World FactBook – пример распределенной базы данных в Web [14], использующей онтологии. Данные в системе The World FactBook формируются Central Intelligence Agency US для правительства США на основании различных источников и баз данных. При интеграции баз данных используются онтологии. В The World FactBook

представлена географическая, демографическая, историческая и экономическая информация о странах мира.

3.5. Системы, поддерживаемые DARPA

Многие системы, работающие с онтологиями (включая перечисленные ранее), поддерживаются DARPA и созданы благодаря финансированию из этого ведомства в больших размерах.

Система Cyc в последнее время частично открывается и переводится на коммерческую основу.

Особое внимание DARPA уделяется обеспечению взаимодействия систем в Интернет и стандартам межмашинного взаимодействия (KIF, OWL).

3.6. Другие примеры разработок онтологий

Онтологии верхнего уровня [15]. Примеры:

• Descriptive Ontology for Linguistic and Cognitive Engineering (DOLCE);
• General Formal Ontology (BFO);
• WordNet;
• Suggested Upper Merged Ontology (SUMO).

Онтологии верхнего уровня – это особая тема. Под онтологией верхнего уровня специалистами, занимающейся этой темой, понимается самая общая онтология, которая является общей для всех областей знаний. Считается, что такая онтология существует, и разные группы специалистов предъявляют свои варианты. Общие онтологии (онтологии верхнего уровня) связаны с мировоззрением, и, естественно, эта область близка к исследованиям в философии и лингвистике. Возникновение нескольких онтологий верхнего уровня и конкуренция в этой области означает, что создание онтологий верхнего уровня не простой вопрос, и требуются удобная (модульная) организация онтологий, ясное представление онтологий, специальные средства, поддерживающие процесс согласования онтологий. Становится очевидным, что разработка онтологий верхнего уровня – это не разовая акция, а постоянный процесс, и нужно иметь инструменты, поддерживающие этот процесс и коллективную работу над онтологиями.

Специализированные онтологии. Примеры:

Список ссылок на онтологии, разработанные с помощью Protégé, приведен в [16]. Имеется большое количество отологий и их применений в биологии и медицине.

Онтологии в корпоративных системах [17].

В больших (распределенных) корпоративных системах онтологии могут использоваться в трех целях:

• для унификации ведущихся в корпорации документов и сбора на их основе данных для ввода в базу данных корпорации;

• для представления и организации метаинформации в системах типа «хранилища данных» с целью использования ее при формировании запросов для экономического анализа данных работы корпорации;

• для ведения, поиска и организации нормативно-справочной информации.

В некоторых больших российских корпорациях, например, «Интегра», «Татнефть», «Норникель», «Сибур», ТНК-ВР уже созданы и используются онтологии для нормативно-справочной информации (фирма проектировщик: НЦИТ ИНТЕРТЕХ, система ONTOLOGIC).

Онтологии и СУБД

В некоторых системах управления базами данных, например ORACLE [18], вводятся специальные средства для работы с онтологиями и для их использования в языке запросов.

3.7. Инструменты для работы с онтологиями

В этом подразделе хотелось бы обсудить инструменты, с помощью которых строятся онтологии, в частности, в виде OWL файлов. Таких инструментальных систем становится все больше с разными претензиями на удобство и универсальность.

Одна из первых разработок в этой области – это система Protégé [19] с большим опытом применения. Судя по тому, как строятся многие онтологии в виде файлов, Protégé является наиболее распространенным инструментом. Система Protégé разработана в лаборатории KSL Стэнфордского университета. Первоначально она разрабатывалась как программное инструментальное средство для формирования словарей в области медицины, но оказалась полезной для применений и в других областях. Protégé 2000 разработана уже для работы в Web-браузерах. В настоящее время с ее помощью читаются и формируются OWL-файлы. На конференции в Будапеште (июль 2007г.) [20] определены проблемы и некоторые направления развития Protégé. Другая система, Chimaera [20], (также разработка подразделения KSL Стэнфордского университета) предназначена для программной поддержки процесса объединения больших онтологий. Это графический редактор, который выделяет сомнительные места в объединенной онтологии и позволяет редактировать онтологию.

4. Проблемы формирования и использования библиотек онтологий

Пятнадцать лет – большой срок в области информационных технологий, и за эти пятнадцать лет технологии построения и использования онтологий становятся яснее. Однако темпы внедрения онтотехнологий все-таки медленны. Главная причина такого замедления является то, что онтологии должны строиться высоко квалифицированными специалистами в своей области, а языки представления онтологий являются сложными, техничными и далекими от этих областей знаний.

Для формирования простейших онтологий в виде классификаций были построены графические редакторы, которые упрощают работу с такими онтологиями и делают их более наглядными. Это определило активность построения классификационных онтологий во многих областях знаний. В свою очередь, когда специалисты стали активно строить классификационные онтологии, у некоторых специалистов стали появляться потребности и в более сложных онтологиях, в библиотеках онтологий, в новых методологиях их построения. Так, например, появились работы [22] об использовании алгебраического языка спецификаций CASL в проекте DOLCE.

Другая причина отсутствия массового использования онтотехнологий в Веб в настоящее время состоит в том, что массовый пользователь не видит непосредственного эффекта от использования онтологий, а от него эти технологии требуют больших усилий по семантической разметке той информации, которую он выставляет в Веб. Поэтому для преодоления этого барьера нужно разработать Веб-среды и инструменты, в которых пользователи смогут создавать собственные семантически размеченные страницы и языки запросов к ним с тем, чтобы пользователи сами могли создавать системы с новой функциональностью в виде информационных систем.

Движение в этом направлении предложено в проекте Semantic Wiki [23].

Понимание проблем приходит с опытом. Ниже перечисляются некоторые, как мне кажется, важные положения современного состояния разработки библиотек онтологий и использования онтологий.

Проблемы формирования и использования библиотек онтологий:

1. Так как онтология есть фиксация в формальном виде договоренностей группы специалистов в определенной области о системе используемых ими понятий, их свойствах и аксиомах, то каждая система онтологий имеет смысл только для группы людей, принимающих эти договоренности (социальный характер онтологий). Поэтому должна

быть обеспечена возможность формирования онтологий для различных групп специалистов.

2. Так как науки и представления в областях знаний меняются, то в компьютерных системах онтологий требуются средства поддержки целостности и версионности онтологий при изменениях и постепенном накоплении онтологий.

3. Так как в онтологиях фиксируются договоренности специалистов, представлять онтологии должны специалисты в предметных областях. Поэтому язык представления онтологий должен быть удобен для этих специалистов. Заметим, что в каждой области знания при формировании понятий этой области формируются и специализированные языки. Поэтому язык представления онтологий должен быть открытым для пользователей. При этом внутреннее представление онтологий для компьютерного использования и межмашинного обмена должно быть стандартизованным.

Проблемы реализации:

1. Большие онтологии и большие библиотеки онтологий требуют разработки специальных средств их ведения.
2. Формирование сложных систем онтологий требует соответствующих средств опробования и отладки онтологий.
3. Для сложных онтологий полностью отделить непроцедурные и процедурные знания не удается (эффективность использования онтологий, прагматика).
4. Модульность построения библиотек онтологий. При этом следует учитывать контекстность онтологий (их возможную взаимную противоречивость), целевое создание и многоцелевое, многоразовое использование.
5. Проблема интеграции онтологий, представленных на разных языках в разных логиках и моделях.

Направления преодоления трудностей формирования и использования больших библиотек онтологий:

1. Использование Web 2.0-технологий в Web для создания социальных сетей и сред в Web, наполняемых самими пользователями (яркий пример – Wikipedia).
2. Построение систем с открытым языком пользователя и стандартным языком внутреннего представления онтологий.
3. Предоставление пользователям Web, при формировании своих страниц, удобных средств модульного (с использованием чужих модулей) формирования внутреннего (семантического) представления данных своих страниц и **языка запросов к странице** (новые сервисы).
4. Алгебраический подход к моделированию онтологий, как путь выработки принципов интеграции разнородных онтологий.

5. Wiki-технологии для формирования и использования библиотек онтологий

Wikipedia – это грандиозное достижение современности. Ее развитие происходило в русле основных принципов Интернет и Веб. Основной принцип Интернет – это распределенность ресурсов и отсутствие единого центра управления. Основной принцип Веб – наполнение содержания сети происходит самими пользователями сети Интернет. Оба эти принципа выдерживаются в Wikipedia и дополняются принципами демократической самоорганизации людей, наполняющих Wikipedia содержанием, а также возможностью использования любой страницы Wikipedia, как шаблона страницы для вашей статьи.

В чем достоинства технологий Wikipedia для создания и использования библиотек онтологий? Первый и очевидный – это социальный характер Wikipedia. Второй, и очень

важный, – это возможность семантически разметить только страницы-шаблоны, освобождая остальных пользователей от тяжкой работы семантической разметки своих страниц. Например, можно создать шаблон для страницы «Person» с соответствующими полями и конкретную страницу некоторой персоны. Другие пользователи, используя эту страницу и редактируя ее, заводят информацию о многих других персонах. Кто-то может завести страницу о конкретном правителе России, добавив на страницу дополнительные семантически размеченные поля (например, годы правления и тип правления). Кто-то, уже используя эту страницу, заведет страницу о конкретном царе из династии Романовых, введя дополнительное поле «династия» и ее значение. Далее, на основе этого шаблона могут быть заведены страницы других Романовых.

Так в Wikipedia могут появиться семантически размеченные страницы. Но эта разметка бессмысленна, если нет языка запросов, для ответов на которые используется данная семантическая разметка.

В качестве языка запросов низкого уровня может использоваться язык SPARQL Query Language for RDF [24], предлагаемый в проекте Semantic Web. На основе этого языка уже может формироваться язык более высокого уровня для пользователей. Пример такого языка имеется в Semantic MediaWiki [25].

Более того, на основе онтологии «Person» в Wikipedia можно создать страницу «Родственные отношения», в которой будут введены шаблоны запросов «Брат», «Сестра», «Дядя», «Теща» и т.д. с соответствующими формулами запросов. Аналогично, можно создать страницу «Престолонаследник» с соответствующим шаблоном и формулой запроса. Наконец, мы можем создать страницу «Династия Романовых», в которой будет присутствовать стандартная текстовая и графическая информация, а часть страницы будет представляться ответом на запрос, в виде таблицы (или дерева) обо всех персонах, страницы которых есть в Wikipedia, и относящихся к роду Романовых. При этом может быть загружены онтологии страниц не только из этих страниц, но и онтологии страниц «Родственные отношения», «Престолонаследник». Так будет сформирована онтология страницы «Династия Романовых», и на ней будут доступны шаблоны запросов страниц «Person», «Родственные отношения», «Престолонаследник», комбинируя которые пользователь может строить сложные содержательные запросы к странице «Династия Романовых». На этой странице могут быть приведены примеры сложных и простых запросов, по аналогии с которыми пользователь может сформулировать свои запросы.

6. Выводы

• Онтосистемы и онтопроекты создаются и развиваются уже более 10 лет. Успех и значимость этого направления очевидны.

• Однако, темп внедрения онтотехнологий все еще невелик. Пока практические успехи получены при финансовой поддержке государственных органов, либо внутри больших корпораций.

• Для широкого внедрения онтотехнологий предлагается строить онтосистемы с использованием следующих трех принципов.

Три принципа построения новых баз онтологий

1. Онтологии строятся в стиле Wikipedia с поддержкой модульности, коллективной работы, версий и системы согласований.

2. В системе поддерживается среда открытого языка работы с онтологиями, который формируется самими пользователями, по мере пополнения базы онтологий.

3. Вместе с текстом онтологии в системе формируется внутреннее представление онтологии, которое используется при семантическом анализе выражений языка, при

формировании ответов на запросы к онтологии и ее отладке, при межмашинном обмене онтологиями в некотором стандарте и при использовании онтологий в приложениях.

В январе 2008 года мной сформирован открытый проект http://ezop-project.wiki.sourceforge.net/ по созданию макета такой системы на основе перечисленных выше принципов построения. Приглашаю всех желающих к обсуждению проекта и к участию в нем.

Литература

1. Gruber T. R. The role of common ontology in achieving sharable, reusable knowledge bases. In J. A. Allen, R.Fikes, and E. Sandewell, editors, Principles of Knowledge Representation and Reasoning – Proceedings of the Second International Conference, pp. 601-602. Morgan Kaufmann (1991)
2. Бениаминов Е.М. Основания категорного подхода к представлению знаний. Категорные средства. // Изв. АН СССР Техн. кибернетика, №2, 21-33 (1988)
3. Бениаминов Е.М., Болдина Д.М. Система представления знаний Ontolingua - принципы и перспективы // НТИ. Сер.2. № 10 (1999)
4. Кожара В.Л. Классификационное движение. // Институт биологии внутренних вод им. И.Д. Папанина РАН, Борок, 2006, http://ibiw.ru/win/kd2.pdf
5. Matuszek C., Cabral J, Witbrock M., DeOliveira J. An Introduction to the Syntax and Content of Cyc, http://www.cyc.com/doc/white_papers/AAAI06SS-SyntaxAndContentOfCyc.pdf
6. Википедия о системе Cyc, http://en.wikipedia.org/wiki/Cyc
7. Википедия о Дугласе Ленате (авторе проекта Cyc), http://en.wikipedia.org/wiki/Douglas_Lenat
8. Официальный сайт компании Cycorp, http://cyc.com/
9. OpenCyc – открытый фрагмент онтологии Cyc, http://www.opencyc.org/
10. Представление онтологии OpenCyc в Web, http://www.cycfoundation.org/concepts
11. Алексеева М. В. Обзор системы Cyc. М.:РГГУ (2008), http://ezop-project.wiki.sourceforge.net/Alekseeva_Cyc
12. Сервер онтологий Ontolingua, http://www.ksl.stanford.edu/software/ontolingua/
13. Примеры использования системы Ontolingua, http://www.ksl.stanford.edu/htw/htw-demos.html
14. The World FactBook, https://www.cia.gov/library/publications/the-world-factbook/index.html
15. Википедия об онтологиях верхнего уровня, http://en.wikipedia.org/wiki/Upper_ontology_(computer_science)
16. Список ссылок на онтологии, разработанные с помощью Protégé, http://protegewiki.stanford.edu/index.php/Protege_Ontology_Library
17. Гладун А.Я., Рогушина Ю.В. Онтологии в корпоративных системах. // "Корпоративные системы" , №1, С. 41-47 (2006), http://www.management.com.ua/ims/ims115.html, http://www.management.com.ua/ims/ims116.html
18. Oracle® Database Semantic Technologies Developer's Guide 11g Release 1 (11.1) Part Number B28397-02, http://download-uk.oracle.com/docs/cd/B28359_01/appdev.111/b28397/toc.htm
19. Protégé, http://protege.stanford.edu/,
20. Конференция в Будапеште (июль 2007г.), http://protege.stanford.edu/conference/2007/schedule.html
21. Chimaera, http://www.ksl.stanford.edu/software/chimaera/

22. Luettich, K.; Masolo, C.; Borgo, S. Development of Modular Ontologies in CASL.// In Proceedings of International Workshop on Modular Ontologies (WoMO), Athens (Georgia, USA), 05 November 2006, http://www.loa-cnr.it/Publications.html
23. Википедия о Semantic Wiki, http://en.wikipedia.org/wiki/Semantic_wiki
24. SPARQL Query Language for RDF, http://www.w3.org/TR/rdf-sparql-query/
25. Semantic MediaWiki, http://semantic-mediawiki.org/wiki/Semantic_MediaWiki
26. Пример системы с запросами на естественном языке, http://www.trueknowledge.com/

Тезисы по высшей онтологии (upper ontology)

Анатолий Левенчук Anatoly Levenchuk
ailev@asmp.msk.su
http://ailev.livejournal.com/profile
http://ailev.livejournal.com/ Лабораторный журнал в Живом журнале http://livejournal.com/

1. Высшая онтология (upper ontology, http://en.wikipedia.org/wiki/Upper_ontology_(computer_science)) - это наиболее общие концепты, уточнением которых можно получить частные концепты конкретных предметных областей. Высшая онтология включает в себя
- мереологию (mereology), определяющую отношения частей и целого;
- топологию (topology), определяющую пространство и время;
- указания на существование реальных и абстрактные объектов и прочие "философские категории", определяющие картину мира.

2. Примеры публично доступных высших онтологий см. в Key Public Upper Ontology Custodians на стр. http://ontolog.cim3.net/cgi-bin/wiki.pl?UpperOntologySummit (разумеется, это неполный список: некоторые специальные онтологии включают в себя и высшую онтологию, и правила расширения, например, OntoCAPE для моделирования в обрабатывающей промышленности, http://www.lpt.rwth-aachen.de/Research/OntoCAPE.php). Эти онтологии не слишком зависят от выбора формального языка их представления, большинство из них могут быть отконвертированы в OWL (для чего каждая из групп-разработчиков по факту сделала конвертер), чтобы непосредственно работать с софтом, обильно разрабатываемым в рамках инициативы Semantic Web (тьюториал Semantic Web: http://www.w3.org/Consortium/Offices/Presentations/RDFTutorial/Slides.pdf).

3. Чтобы разрабатывать собственные промышленные приложения, нужно быть уверенным, что эти приложения смогут добыть/положить данные из/в другие приложения. Для этого нужно иметь возможность описать целостную структуру собственных данных и уметь обратиться к таким же описаниям в других приложениях. Совместимые описания получаются тогда, когда в них используется общий взгляд на мир - а это получается при условии использования одной и той же высшей онтологии. Тем самым первая задача при разработке промышленного приложения - это выбор конкретной высшей онтологии.

4. Одна из высших онтологий (примерно 200 понятий) была сконструирована европейским консорциумом EPISTLE (http://www.infowebml.ws/) специально для задач

долгосрочной интеграции данных компьютерных приложений обрабатывающей промышленности, и зафиксирована комитетом ISO TC184/SC4-Industrial data в виде стандарта ISO 1592-2:2003. Расширение этой онтологии известно как Reference Data Library и включает в себя около 20000 концептов из предметной области инжиниринга (в онлайне можно поглядеть тут: http://projects.dnv.com/reference_data/RD7Browser/). Первоначально (по стандарту) эта онтология была записана на языке EXPRESS (ISO 10303-11, 1994), а затем оттранслирована на язык OWL (http://www.matthew-west.org.uk/Documents/batres-final.pdf).

5. Высшая онтология ISO 15926-2:2003 основана на картине четырехмерного мира: три пространственных размерности и время как четвертая размерность. В четырехмерном мире объекты протяженны во времени так же, как в пространстве. Особенно это важно при описании объектов на всем протяжении их жизненного цикла, включая сборку, функционирование с ремонтом, а затем разборку объектов - в том числе и таких сложных объектов, как заводы, корабли и самолеты, электростанции.

6. С набором стандартов ISO 15926 активно работает промышленность, очень много сайтов поддерживающих проектов, Shell Petroleum обязала своих поставщиков оборудования использовать именно этот стандарт для передачи информации при закупках. Есть несколько промышленных консорциумов, занимающихся обеспечением совместимости компьютерных приложений с использованием этого стандарта (см. примеры в http://ailev.livejournal.com/515742.html). Современная справка о состоянии дел -- http://www.infowebml.ws/Topics/papers/15926SW.htm (на 7 августа 2007г.).

6. Из набора стандартов, в которых также есть высшая онтология нужно особо выделить стандарт описания процессов (process specification language, PSL, http://www.mel.nist.gov/psl/projects.html) ISO 18629, который подготовила объединенная рабочая группа подкомитетов 4 (промышленные данные) и 5 (интеграция производства) технического комитета 184 (промышленные системы автоматизации и интеграция). Назначение этого стандарта быть не языком моделирования процессов (их и так хватает), а быть языком обмена между системами, моделирующими процессы, например, обеспечивать обмен данными между приложениями на IDEF3 и приложениями на сетях Петри. Пока среди разработчиков и пользователей стандарта PSL много больше академических ученых, нежели представителей промышленности. Зато у них очень тесные связи с группой разработчиков Semantic Web в W3C.

7. Высшие онтологии жестко конкурируют друг с другом, но среди них есть горячее желание объединиться. С этой целью ежегодно проходят встречи хранителей (custodians) этих онтологий, последняя встреча была в апреле 2007г. - http://ontolog.cim3.net/cgi-bin/wiki.pl?OntologySummit2007. Один из результатов этой встречи -- появление текста про различение разных (компьютерных) онтологий: http://ontolog.cim3.net/cgi-bin/wiki.pl?OntologyDistinctions, ибо для объединения нужно как минимум разобраться, что именно объединяется (ибо кроме "классических" онтологий есть еще таксономии, фолксономии, тезаурусы и т.д. - все они в чем-то похожи).

8. Рабочие встречи идут непрерывно, вот пример их организации: http://ontolog.cim3.net/cgi-bin/wiki.pl?ConferenceCall_2007_10_11 (как раз посвящен распределенной разработке онтологий - в данном случае 50000 концептов медицинской информации, которые редактируются с удаленных рабочих мест по всему миру), http://ontolog.cim3.net/cgi-bin/wiki.pl?ConferenceCall_2007_10_04 (что нужно от систем поддержки коллаборативной разработки онтологий). Похоже, что все системы моделирования - это системы коллаборативной разработки онтологий, и этот опыт через

десяток лет будет мейнстримом вообще распределенной разработки чего бы то ни было.

9. Современные онтологии подразумевают не только описание понятий, но и описание их связей (правила). Поразвлекайтесь с моделью Чернобыльского реактора http://www.ida.liu.se/~her/npp/demo.html - эта демонстрашка работает на базе правил, задаваемых в редакторе онтологий Protege (http://protege.stanford.edu/plugins/powerplant/powerPlant_screenshot.html). Вот страничка этого, пожалуй, самого известного на сегодня редактора онтологий: http://protege.stanford.edu/

10. В 2006-2007г.г. в онтологиях случился таки прорыв, количество перешло в качество, и онтологический подход тихо и незаметно вошел в промышленный мейнстрим в его стремлении интегрировать все и вся -- и вошел этот подход как раз через промышленные международные стандарты ISO TC184.

11. И используются сейчас онтологии по любому поводу (например, при теоретизировании предприятия-как-системы онтология тоже поминается между law и language: http://www.enterprisesystemtheory.net/index.html). И напоследок -- еще одна инициатива, которая так и называется Universal Business Language 2.0. Вот он, универсальный бизнес по версии OASIS (смотреть нужно прелестную картинку в 4-м разделе): http://docs.oasis-open.org/ubl/os-UBL-2.0/UBL-2.0.html. Похоже, что онтологический мир разделился на мир синих воротничков и мир белых воротничков. Мне мир синих воротничков нравится много больше. Поэтому высшую онтологию я буду выбирать такую, в которой по факту описано максимальное количество *вещей*. Похоже, что это ISO 15926. А уж процессы "обеспечения качества" или "управления проектами и программами" - это мы еще посмотрим, в какой высшей онтологии их описывать.

Компоненты онтологических систем и их реализация в современных проектах

Л. М. Пивоварова, В. Ш. Рубашкин
Санкт-Петербургский государственный университет, Санкт-Петербург

«Интернет и современное общество» (IMS–2007), с.277-279

Данная работа является обзором основных точек зрения на представление знаний и управление знаниями в онтологических системах. В настоящее время уже существует несколько обзоров онтологических систем (см., например [1][12][6][4]). Помимо того, что подобные обзоры быстро устаревают, все они различаются как по своим исходными позициями, так и по выбору характеристик, которые авторы считают существенными при сопоставлении различных систем. Мы считаем наиболее важными следующие аспекты:

- Базовый язык представления знаний и аксиоматика онтологии.
- Наполнение онтологии знаниями и интеграция накопленных ранее знаний.
- Ограниченный логический вывод.
- Связь онтологии с естественным языком.
- Инструментарий для работы с онтологиями (онторедактор).

Хотя все перечисленное важно для создания полнофункциональной онтологической системы, реальные проекты нередко сосредотачивают внимание только на некоторых из этих пунктов.Далее рассматриваются четыре крупных и, по-видимому, наиболее авторитетных онтологических проекта: Protégé [11], SUMO (Suggested Upper Merged Ontology) [13], KAON2 (The KArlsruhe ONtology and Semantic Web tool suite 2) [5] и CYC [2].

Эти системы по-разному функционально ориентированы, и их трудно сравнивать по единым критериям. Если говорить об основной функциональной ориентации, то Protégé — онторедактор; SUMO — универсальная онтология; KAON (далее рассматривается последняя версия KAON2) — система логического вывода и ответов на запросы к концептуальной системе, CYC — технология, направленная на поддержку бизнес-процессов и создание интеллектуальных агентов. В то же время, разнонаправленность этих систем позволяет осветить различные аспекты инженерии онтологий.

Protégé — это онторедактор и инструмент для построения баз знаний, созданный на медицинском

факультете стенфордского университета группой медицинской информатики (Stanford Medical Informatics) под руководством профессора Марка Мьюсена [10].

Protégé не имеет никакой встроенной онтологии и рассчитан на создание прикладных онтологий с нуля. Более того, разработчики Protégé считают методически правильным начинать разработку онтологии, «подвешивая» понятия прикладного характера (***Пицца***, ***Страна***, ***Вино***) непосредственно, как подклассы понятия ***Нечто*** [4][9], и тем самым фактически игнорируют проблему общей организации концептуальной модели. Связь с естественным языком также не предусмотрена, хотя Protégé позиционируется как встраиваемый компонент для работы в системах извлечения знаний [6].

Несмотря на эти недостатки, Protégé активно используется в различных прикладных проектах, т. к. позволяет быстро и относительно просто сконструировать небольшую предметную онтологию и имеет расширяемую архитектуру, которая позволяет легко встраивать его в прикладные программы.

Языком описания онтологии в Protégé является OWL (либо еще более упрощенная фреймовая структура).

Соответственно, могут быть реализованы основные выразительные возможности OWL — различение индивидов, классов и свойств; иерархия классов и свойств; указание области применения и области значений для объектных свойств и др. Индивид может принадлежать нескольким классам одновременно или не принадлежать ни одному классу. Также существует возможность указать, что пара или группа классов являются взаимоисключающими.

Классы могут не только описываться, но и определяться через указание его необходимых и достаточных свойств.

SUMO (Suggested Upper Merged Ontology, Рекомендованная онтология верхнего уровня интеграции) — это бесплатная, свободно распространяемая онтология, принадлежащая IEEE (Institute of Electrical and Electronics Engineers). SUMO не предназначена для решения какой-либо конкретной задачи и преследует цель интеграции существующих онтологий в единую, всеобъемлющую, способную к расширению структуру, которая имела бы статус универсального стандарта и могла бы использоваться в различных прикладных и исследовательских проектах [8].

SUMO является онтологией верхнего уровня и содержит около 1000 понятий и 5000 аксиом, из них около 800 правил. Более конкретные понятия хранятся в отдельных отраслевых онтологиях и могут подключаться по мере необходимости. Для связи между SUMO и отраслевыми онтологиями разработана онтология среднего уровня MILO (Mid-Level Ontology). Весь доступный на сегодня комплекс онтологий содержит около 20000 понятий и 60000 аксиом [13].

Самым общим понятием SUMO является ***Сущность*** (***Entity***). Сущности разделяются на ***Физическое*** и ***Абстрактное***. К категории ***Абстрактного*** в онтологии SUMO относятся такие понятия как ***Количество***, ***Атрибут***, ***НаборИлиКласс***, ***Отношение***, ***Суждение***, ***Граф*** и ***ЭлементГрафа***. ***Физическое*** разделяется на ***Объекты***, ***Процессы*** и ***НосителиСодержания*** (***ContentBearingPhysical***). В онтологии SUMO допускается множественное наследование [8].

Изначально SUMO разрабатывалась на диалекте языка KIF, в настоящее время выполнен ее перевод на OWL. Разработана полная система соответствий понятий SUMO с лексическими единицами WordNET (для английского языка). SUMO является онтологией в чистом виде и не имеет ни достаточно развитого онторедактора, ни машины логического вывода; создатели SUMO предоставляют лишь информацию, которая может обрабатываться программно и включаться в качестве составной части в различные приложения и обрабатываться средствами этих приложений.

KAON2 (The KArlsruhe ONtology and Semantic Web tool suite 2) — это инфраструктура для управления онтологиями, созданная в университете Карлсруэ при сотрудничестве с Манчестерским университетом.

KAON2 позволяет управлять онтологиями, имеет программный интерфейс для взаимодействия с другими приложениями, может использоваться в качестве онтологического сервера, обладает машиной логического вывода и механизмом извлечения знаний из реляционных баз данных [5]. Система KAON2 также не разрабатывалась под какую-либо конкретную задачу, однако интересы ее разработчиков лежат в области Semantic Web и бизнес-процессов.

Ни проект KAON2, ни другие проекты, осуществляемые в группе по управлению знаниями университета Карлсруэ, не ставят своей целью разработку онтологий, т. е. непосредственно концептуализацию. Создателей KAON'а интересует не столько содержание онтологии, сколько ее формальная структура. С точки зрения структуры, принципиальным является разделение онтологии на терминологическую часть (TBox) и часть данных (ABox, «A» означает «assertions», утверждения). Поскольку проект ориентирован, главным образом, на разработки в области Semantic Web, наиболее значимыми оказываются онтологии с компактной T-частью и большой A-частью (большое количество документов аннотируется на основе относительно простой онтологии [7]).

С вычислительной точки зрения основной функцией KAON2 является логический вывод и ответы на запросы о составе и свойствах концептуальной системы, поэтому модель знаний KAON2 наиболее детально проработана и единственная из всех рассмотренных в данном обзоре имеет логико-теоретическое обоснование.

KAON2 обладает интерфейсами для импорта онтологий на языках OWL-DL, SWRL и F-Logic, однако для задач логического вывода используется собственный внутренний язык, основанный на клаузах Хорна.

В процессе ответа на запросы T-часть онтологии переводится в программу логического типа, которая затем исполняется, используя A-часть (аксиомы) в качестве данных [7].

CYC [3] — это многоцелевая база знаний и машина логического вывода, направленная на семантизацию программного обеспечения за счет базовых интуитивных знаний (здравого смысла). CYC является коммерческой технологией, ориентированной, в первую очередь, на использование в бизнес-процессах — для таких задач как интеграция разнородных баз данных, интеллектуальный поиск, распределенный искусственный интеллект и др.

База знаний Cyc описывается на языке CycL, который является гибридным языком, сочетающим в себе свойства фреймов и логики предикатов [4]. CycL различает индивиды, классы, предикаты и функции. Предикаты могут иметь любое количество аргументов, при этом основное содержание онтологии — это предложения, сконструированные из

предикатов и представляющие некоторые сведения о реальном мире. Предпочтение отдается не столько фактам (предполагается, что их интеллектуальный агент получает из баз данных, Интернета и других источников), сколько знаниям, необходимым для их понимания [10]. Различие между коллекциями и индивидами определяется лишь применяемыми к ним предикатами: коллекции выстраиваются в иерархию с помощью предиката ***IS_A***, а индивиды связываются с коллекциями через предикат ***быть экземпляром*** [2].

На глобальном уровне к онтологии не предъявляется требования непротиворечивости — поскольку «наивные» человеческие представления, которые она призвана отражать, также противоречивы. Вместо этого онтология разбивается на отдельные микротеории, которые выстраиваются в иерархию, при этом число уровней иерархии достигает 50. Все утверждения в микротеории опираются на некоторые общие допущения и не должны противоречить друг другу.

Верхний уровень онтологии содержит микротеории, описывающие индивиды, коллекции, логические отношения; далее — пространство, время, движение, действие и т. п. Средний уровень — астрономия, физические объекты, химия, география, погода, экология, растения и животные, социальное поведение, организации, политика, бизнес, военное дело и др. Еще ниже расположены доменные минитеории. В настоящее время онтология CYC содержит около 400 тысяч концептов и более 3,5 миллионов утверждений [2].

Важнейшим компонентом базы знаний CYC является **лексикон**, также построенный на основе WordNET.

CYC не является онтологией значений слов: не все концепты имеют ссылку на обозначающее их слово и, наоборот, не все значения слов лексикона отображены в онтологии. Концепты (значения) добавляются в онтологию только в том случае, если это необходимо для поддержки логического вывода на основе здравого смысла. Хотя содержательно лексикон сильно отличается от онтологии, с формальной точки зрения он является никак не выделяемой частью базы знаний и описывается на том же языке CycL, что и основная онтология.

Таким образом, можно констатировать, что крупные онтологические проекты с точки зрения их функциональности не столько конкурируют, сколько дополняют возможности друг друга. Одновременно можно, видимо, констатировать, что пока не существует проекта, который мог бы использоваться как готовый и наполненный знаниями полнофункциональный ресурс, готовый как для непосредственного использования, так и для дальнейшего развития в широком спектре интеллектуальных информационных технологий.

Работа выполнена при финансовой поддержке РФФИ (проект № 06-06-80434).

Литература

1. *Овдей О. М.,* Проскудина Г. Ю. Обзор инструментов инженерии онтологий // Труды Шестой Всероссийской научной конференции «Электронные библиотеки: перспективные методы и технологии, электронные коллекции», 2004

2. *Cyc* 101 Tutorial [Электронный ресурс]. — Режим доступа: http://www.opencyc.org/doc/tut/

3. *Cycorp,* Inc. [Электронный ресурс]. — Режим доступа: http://www.cyc.com

4. *Gomez-Perez* A., Fernando-Lopez M., Corcho O. Ontological Engineering with examples from the areas of Knowledge management, e-Commerce and the Semantic Web – 2nd. Ed. – London, Springer-Verlag – 2004

5. *KAON2* — Ontology Management for the Semantic Web [Электронный ресурс]. — Режим доступа: http://kaon2.semanticweb.org/.

6. *Mizoguchi R.* Ontology Engineering Environments // Handbook on Ontologies; ред. Staab S., Studer R. – Berlin, Springer-Verlag — 2003

7. *Motik B.,* Sattler U. A Comparison of Reasoning Techniques for Querying Large Description Logic ABoxes //Proc. of the 13th International Conference on Logic for Programming Artificial Intelligence and Reasoning (LPAR 2006) — Phnom Penh (Cambodia) — 2006

8. *Niles, I.,* and Pease, A. 2001. Towards a Standard Upper Ontology. // Proceedings of the 2nd International Conference on Formal Ontology in Information Systems (FOIS-2001); ред Welty C., Smith B — Ogunquit (USA) — 2001.

9. *Noy, N. F.,* McGuinness, D. L. Ontology Development 101: A Guide to Creating Your First Ontology. Stanford Knowledge Systems Laboratory Technical Report - 2001.

10. *Panton K.* et. al Common Sense Reasoning — From Cyc to Intelligent Assistant // Ambient Intelligence in Everyday Life, pp. 1-31, LNAI 3864, Springer, 2006.

11. *The* Protégé Ontology Editor and Knowledge Acquisition System [Электронный ресурс]. — Режим доступа: http://protege.stanford.edu/.

12. L. Stojanovic, B. Motik Ontology Evolution within Ontology Editors // Proceedings of EON2002 Evaluation of Ontology-based Tools, 2002.

13. Suggested Upper Merged Ontology (SUMO) [Электронный ресурс]. — Режим доступа: http://www.ontologyportal.org/index.html.

The Need for Context in Information Exchange

https://www.posccaesar.org/wiki/ISO15926Primer_Context

•Robin Benjamins, Bechtel Steering Committee Chairman of the POSC Caesar project: Intelligent Data Sets (IDS)
•Onno Paap, Fluor Corporation Project Manager of the FIATECH project: Accelerating Deployment of ISO 15926 (ADI)
•Julian Bourne, NRX Global
•Ian Glendinning, DNV
•Hans Teijgeler, Flour Corporation

Робин Бенджаминс, Bechtel Председатель Руководящего комитета проекта POSC Цезаря: Интеллектуальный наборов данных (IDS)
• Онно Paap, Fluor Corporation Менеджер проекта FIATECH: ускорение развертывания ISO 15926 (ДСП)
• Джулиан Борн, NRX Глобальной
• Ян Глендиннинг, DNV
• Ганс Teijgeler, Мука корпорации

Abstract

This section introduces the concept of Context. When we exchange plant information with traditional methods we rely on context to retain meaning. Information in ISO 15926 format is rich enough that we no longer need context to retain meaning.

How it Used to Be

When your humble author started his career in plant design, computers were not commonly used by designers and engineers. Drafting was by pencil on paper. Specifications were written with a typewriter. When information was transferred from one document to another the only way was for a human to read the original document, find the value to be transferred, then write it by hand on the target document. If the target document was something like a specification it was usually given to a secretary for typing.

Transferring information from one storage location to another was cumbersome, but conceptually simple--you would take all the specifications and drawings, sort them into some logical order, perhaps bind them into books, and move them to the new location. Data turnover to the client at the end of a plant design project was similar to the last scene of the movie *Raiders of the Lost Arc*. In that scene a forklift carried a wooden box down a long aisle of identical wooden boxes and put it on one of the piles. In the real world it sometimes took years for the owner to review all the boxes and categorize the binders of information.

No one really liked this (as in: "I really liked that piece of chocolate cake, may I have another!"), but that was just the way it was. It started to change with computers made their way into the design office. Binders of data sheets gave way to spreadsheets burned onto CDs, graphite pencils gave way to electronic pencils (i.e., CAD), and rolls of mylar drawings gave way to CAD files burned onto more CDs.

Current Situation

There have been improvements, but things haven't changed much conceptually. In our work processes for plant design or plant operations, a large proportion of an engineer's activities still involve manually transferring information from one document to another. For instance, after the engineer chooses, say, an instrument, the only practical way to record the information about the instrument is to read the manufacturer's data, interpret it to decide which of the data values to transcribe, then figure out where to put the data values in the plant design system. Some of the operations are simple transcription, such as transferring a model number from one spot to another. But some involve calculation, such as changing from one unit of measurement to another. Others involve interpretation ranging from ignoring the data value altogether to decisions involving judgment, such as orientation or handedness. The work is done on a computer, but often the only real difference is that engineers do the typing themselves instead of giving it to their secretary.

Why we Need Context

Suppose you have to transfer information from one data sheet to another and you see this:

1034

This means nothing. So you "back up" and look for more context.

Pressure: 1034

OK, so you know a bit more, but still nothing usable.

Pressure: 1034 kPa

Now you expect other values to be in SI units, but you still really don't know what is going on, so you "back up" some more.

Seal Flush
Pressure: 1034 kPa

You still have questions so you continue to "back up".

Tag No: P-101
Service: Chemical Injection to D-101
Seal Flush
Pressure: 1034 kPa

Now you are getting a clearer picture. When you "back all the way up" and read the entire data sheet you can finally put the initial value, 1034, into context.

Centrifugal Pump Data Sheet
Client: ABC Chemical Company
Tag No: P-101
Service: Chemical Injection to D-101
...
Seal Flush
Pressure: 1034 kPa
...

Without context, we are lost.

The Data Sheet Problem

Figure 1 shows sections of two centrifugal pump data sheets. One data sheet might be from a manufacturer's Internet site; the other might belong to the plant owner. It is an engineer's job to interpret the manufacturer's data sheet and transcribe the correct values to the project data sheet.

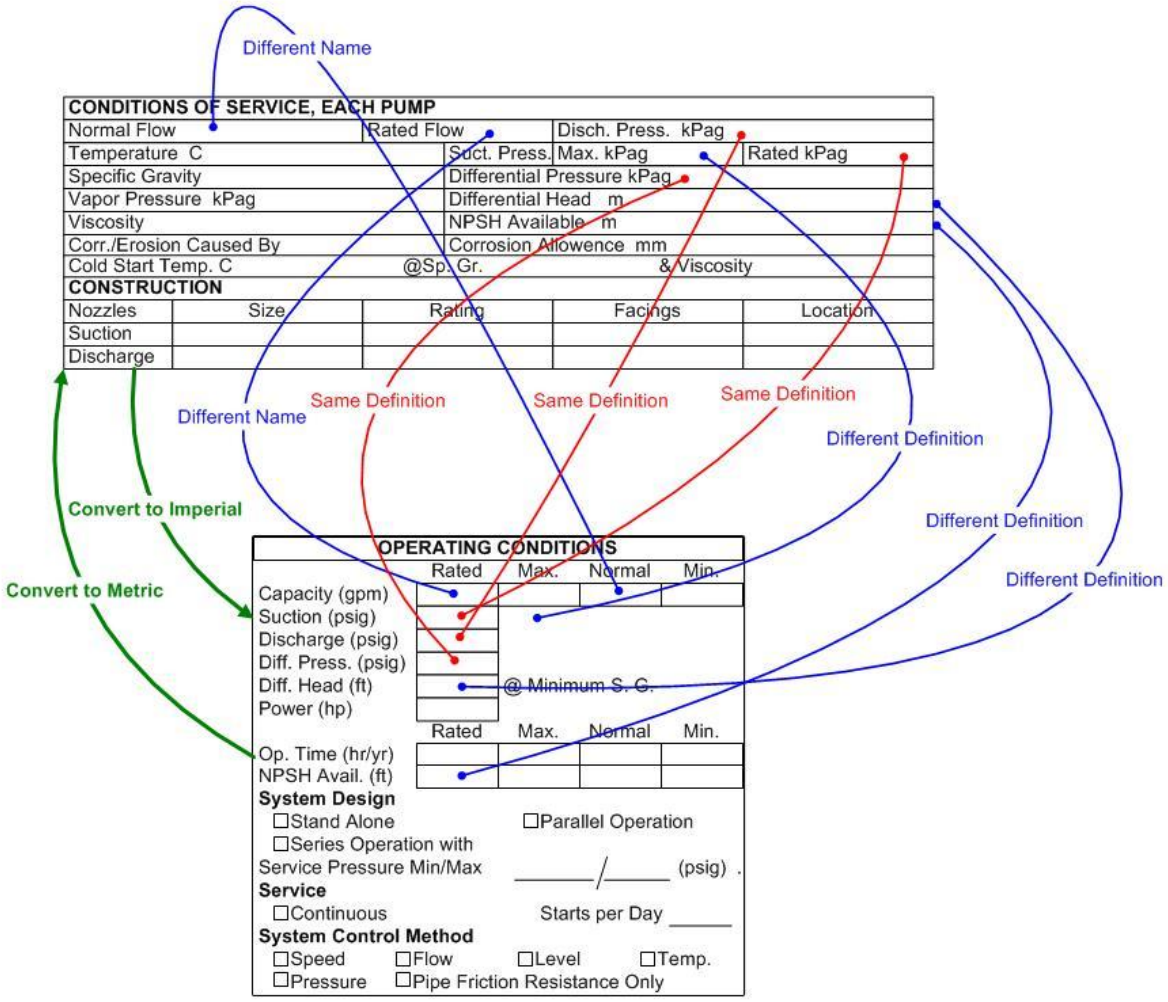

Figure 1: Compare Two Data sheets

The most notable difference is that one data sheet expects Metric units, the other Imperial. But beyond that, the data sheets are organized differently--the data are grouped differently, and

the groups are arranged differently. These two excerpts only have eight data spots in common. But looking closer, of the eight spots, only three are obviously identical:

Discharge Pressure
Rated Suction Pressure
Differential Pressure

The rest require some interpretation:

Metric Data sheet	Imperial Data sheet	Comments
Normal Flow	Capacity: Normal	Probably the same
Rated Flow	Capacity: Rated	Probably the same
Max. kPag	Suction: Max.	No data entry spot
Differential Head	Diff. Head: Rated	Possibly the same
NPSH Available	NPSH Avail.: Rated	Possibly the same

The Challenge

When we exchange information without context we make it difficult for others to understand what we mean.

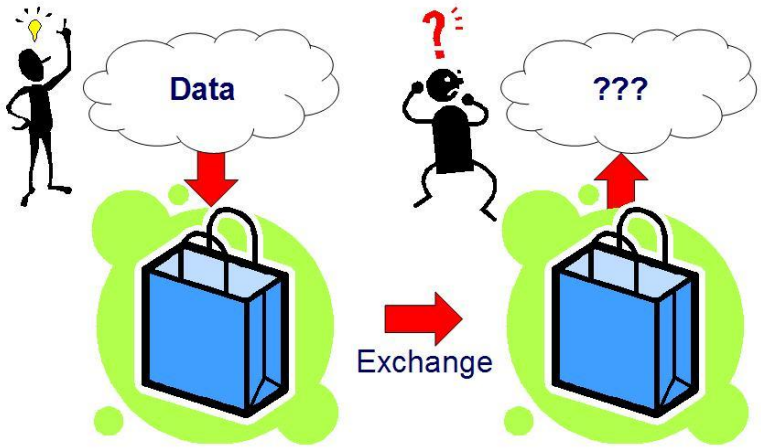

Figure 2 - Putting Information in a Bag

Figure 2 starts with someone having a bright idea. To achieve some business result he has to pass the information to someone else. If he just sends the information without context, he is just throwing it all in a bag, hoping the person on the other end can figure it out.

The reason information exchange worked in the past was that we exchanged entire sets of data (for instance, a complete data sheet) where the context was preserved. But the disadvantage is precisely that: we have to exchange whole sets of data and have humans interpret them item-by-item. What we really want is to be able to let machines exchange information directly without having to rely on context to retain meaning.

Why Can't we Just Cut and Paste?

What we really need is a "cut and paste" tool for plant information. We want to be able to just "cut it from that data base over there" and "paste it to this data base over here". But it's not that simple.

The first and most obvious reason we can't just use a simple Cut and Paste tool is because the data values we want to transfer seldom map to the same (x,y) coordinates on any two data

sheets. In order to know which data values to transfer we have to first know enough about the data sheets and underlaying databases to know which values are involved.

The second reason we can't just use a simple Cut and Paste tool is that mathematical transformations are sometimes required, such as conversion between Metric units and Imperial, as in the data sheet example above. The third reason is that engineering judgment is often required. All of these actions are trivial if you have the right context. We have many thousands of design engineers doing this all day, every day, and generally, they are good at it. But we rely so much on context to convey meaning that we cannot trust machines to make the right decisions on their own.

Using ISO 15926 to exchange information means you no longer have to know anything about your information exchange partner.

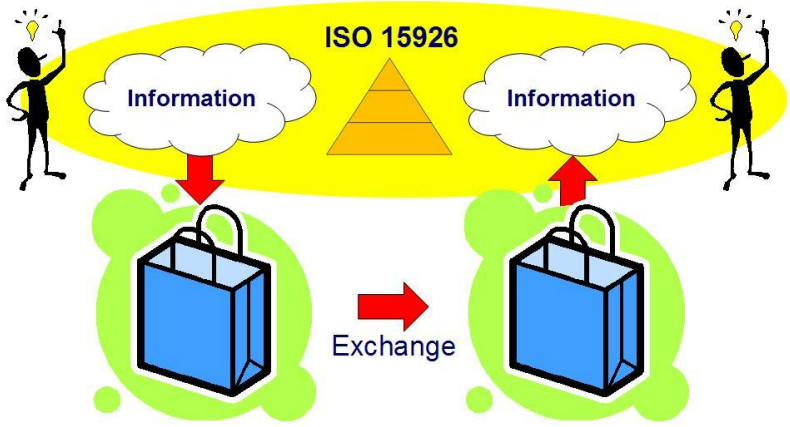

Figure 3 - Putting Information in an ISO 15926 Bag

When we encode information in ISO 15926 format, we include enough context that other ISO 15926-enabled tools will clearly understand what we mean.

Роль онтологий в современной компьютерной науке

Автор: *Лапшин Владимир Анатольевич*
Источник: *RSDN Magazine #4-2009*

Опубликовано: 23.07.2010

Что такое онтология

Мотивация

Хотя термин «онтология» сейчас достаточно популярен в программистском сообществе, четкого его понимания еще не сложилось. Знания о том, что такое онтологии, и как их использовать при создании информационных систем, до сих пор являются чем-то эзотерическим, доступным только избранным специалистам по обработке знаний. Другое мнение состоит в том, что онтологии представляют собой нечто абстрактное, неприменимое на практике «игрушечное знание», которым занимаются в своих «отвлеченных сферах» так называемые «crazy scientists», в просторечье именуемые «ботанами». Между тем, термин «онтология» совсем не сложен для понимания и был придуман для достижения вполне практических целей. В этой статье автор постарается

объяснить, для чего были придуманы онтологии, и как их можно использовать при построении информационных систем.

Конечно, представить в журнальной публикации подробную онтологию использования онтологий в компьютерных системах не представляется возможным. Для полноценного описания всех аспектов такого использования необходима целая книга. В связи с этим автор рад представить свою книгу «Онтологии в компьютерных системах», изданную в 2010 году в издательстве «Научный мир» [1].

Онтологии, упрощенно говоря, представляют собой описания знаний, сделанные достаточно формально, чтобы быть обработаны компьютерами. Такие формальные описания используются в самых различных и порой достаточно неожиданных областях компьютерной науки. Далее мы рассмотрим, какие обстоятельства привели к возникновению термина «онтология», а также опишем некоторые популярные аспекты его использования при написании программ.

Онтологии как интерфейсы интеллектуальных систем

Термин "онтология" впервые появился в работе Томаса Грубера [2], в которой рассматривались различные аспекты взаимодействия интеллектуальных систем между собой и с человеком. Интеллектуальными системами называются программы, которые моделируют некоторые аспекты интеллектуальной деятельности человека. Конечно, любая программа занимается таким моделированием в той или иной степени, ведь именно в этом и состоит ценность компьютера для человека – компьютерная система позволяет освободить человека от выполнения какой-то однотипной деятельности. Эта деятельность может быть весьма сложной и изощренной, но она всегда однотипна: компьютерная система, созданная, например, для редактирования графики, не может быть использована для управления комбайнами во время сенокоса. В этом смысле знания, которые закладывает в программу ее создатель (т.е. алгоритм этой программы), всегда статичны, они не меняются (конечно, за исключением очень конкретных знаний, которые мы называем «данными программы»). Интеллектуальная система в этом смысле более универсальна – в ней знание о том, что надо делать в процессе исполнения программы, не вшито в программу раз и навсегда, но может меняться. Если так, то эти знания необходимо передавать программе как данные, т.е. возникает необходимость их описания.

Знания, которые заложены в компьютерных программах, можно разделить на два сорта:

1. *Процедурные* знания, т.е. знания о том, что надо сделать в каждой конкретной ситуации. Например, если бухгалтерской программе пришли данные о платежах, то надо сделать соответствующие изменения на счетах получателей платежей, а также другие необходимые действия, налагаемые данной ситуацией.
2. Кроме процедурных знаний, каждой программе необходимы знания *о мире задачи* или *декларативные* знания, т.е. о том, что такое платежи, проводки, счета и т.п. вещи. Без этих знаний, очевидно, программа не сможет функционировать, нельзя будет построить алгоритм программной системы.

Таким образом, при создании интеллектуальной системы приходится учитывать такое разделение знаний и придумывать какие-то программные инструменты для оперирования этими знаниями.

Томас Грубер рассматривал вопросы взаимодействия интеллектуальных систем между собой, а также с человеком. Идея Грубера состояла в том, чтобы позволить интеллектуальным системам обмениваться между собой заложенными в них знаниями о мирах задач. Если внутри интеллектуальной системы знания о мире могут быть закодированы как угодно, то для обмена этими знаниями с другой интеллектуальной

системой необходимо предоставить *описание* этих знаний. Это описание должно быть в достаточной степени формальным, чтобы быть понятным другой системе, а также должен быть известен язык этого описания. Кроме того, описание должно быть понятно также и человеку. Для этого Грубер предложил описывать знания двумя способами:

- В канонической форме, которая представляет собой описание знаний на языке логики предикатов (например, в виде фактов языка Prolog).
- В форме *онтологии*, которая представляет собой множество классов, связанных между собой отношением обобщения (это обратное отношение для отношения наследования).

Таким образом, онтология по Груберу представляет собой описание декларативных знаний, сделанное в виде классов с отношением иерархии между ними. К этому описанию, предназначенному для чтения человеком, присоединено описание в канонической форме, которое предназначено для чтения машинами. Каждая интеллектуальная система может предоставлять несколько таких описаний, соответствующих различным областям хранящихся в ней декларативных знаний и, таким образом, выступает как хранилище *библиотеки* онтологий. Грубер представлял, что интеллектуальные системы будут выступать как библиотеки онтологий и свободно обмениваться онтологиями между собой. При этом библиотеке онтологий уже не обязательно быть интеллектуальной системой, достаточно просто предоставлять сервис по передаче онтологий по требованию.

Составление описания декларативного знания обычно требует большой работы и определенных навыков. Для обозначения этой работы, а также ее результата, Грубер ввел в обиход специальный термин «концептуализация». Описание он называл «спецификацией». Таким образом, онтология по Груберу определяется как *спецификация концептуализации*.

Современное понимание термина «онтология»

- Введенное Грубером разделение спецификаций знаний на две составляющие (каноническую форму и онтологию) не очень удобно, т.к. приходится описывать одни и те же знания два раза. Современные языки описания онтологий позволяют совместить эти формы спецификаций в единое целое. Таким образом, сейчас под онтологией понимается любое описание декларативных знаний, сделанное на формальном языке и снабженное некоторой классификацией специфицируемых знаний, позволяющей человеку удобно воспринимать их. Каноническая форма не обязательно использует язык логики предикатов, могут использоваться и другие формализмы. Например, можно использовать т.н. алгебраический подход к описанию знаний [3], в котором факты представляются в виде термов, а различные соотношения между фактами – в виде ограничений, налагаемых на вид этих фактов, и выраженных в форме аксиом эквивалентности. Но любое такое описание должно включать в себя представление декларативных знаний в виде иерархии объектов (классов), только в этом случае это описание может считаться онтологией.
- Некоторые авторы (см. например [4,5]) используют термин «онтология» только для спецификаций знаний о мире, т.е. таких концептуализаций, целью которых является описание структуры Бытия безотносительно какой-либо инженерной задачи. Такой концептуализацией уже давно занимаются философы, и в философии термин «онтология» применяется именно в этом смысле – как спецификация знаний об окружающем мире. Программисты, однако, сталкиваются с иного рода задачами: они проводят концептуализацию с целью построения модели решаемой задачи. Таким образом, философы и программисты преследуют различные цели, когда проводят

концептуализацию: первые имеют целью описание свойств окружающей реальности, а вторые строят формальную модель конкретной задачи. Для философских концептуализаций предлагается использовать термин «онтология», а для концептуализаций инженерных задач – термин «концептуальная схема». Последний термин уже используется в теории баз данных, где обозначает результат построения модели задачи. В реляционных базах данных концептуальная схема представляет собой не что иное, как схему базы данных. Термин «концептуальная схема», который предлагается использовать для обозначения спецификаций концептуализаций программных моделей, представляет собой более широкое понятие. В данной статье не проводится такого разделения и используем термин «онтология» для обозначения любого описания знаний, безотносительно того, с какими целями это описание проводилось.

- Читатель может задаться вопросом: для каких целей вообще может понадобиться построение онтологий в философском смысле, кроме какой-нибудь специфической задачи построения онтологии для некоторой программы философского справочника? Оказывается, спецификации такого рода необходимы для задачи слияния онтологий. Для слияния онтологий необходимо, чтобы эти онтологии были каким-то образом согласованы друг с другом: должна быть согласована терминология, выделены термины, обозначающие одни и те же классы, описываемые факты не должны противоречить друг другу. Только в этом случае можно попытаться слить две разных спецификации в одну. Здесь на помощь приходит философская онтология. Сливаемые онтологии сначала присоединяются к философской онтологии, и на основе этого присоединения происходит согласование сливаемых онтологий. Философские онтологии в абсолютном большинстве случаев описывают очень абстрактные знания, без какой-либо конкретики, поэтому их часто называют онтологиями *верхнего уровня*. Например, в онтологии верхнего уровня придется описать, что такое материальный объект, чем он отличается от нематериального, и тому подобные вещи. Сейчас имеется довольно большое число онтологий верхнего уровня, использующих различные подходы для концептуализации Бытия, с их описанием заинтересованный читатель может ознакомиться в [1].

Онтологии в Интернет

До сих пор повествование было посвящено довольно абстрактным вещам, настало время перейти к чему-то более конкретному. В данном разделе мы рассмотрим, как онтологии применяются для описания содержимого Web-страниц.

Зачем нужно описывать содержимое Web-страницы

Онтологии содержимого Web-страниц необходимы поисковым программам для улучшения качества поиска по Web. Идея построения спецификаций концептуализаций содержания Web-страниц находится в основании концепции так называемого *Умного* Web или *Semantic Web*. Semantic Web представляет собой следующее поколение World Wide Web, в котором кроме гипертекстовых документов содержатся описания семантики этих документов, а также описания семантики различных сервисов, предоставляющих эти документы конечным пользователям. Обычно о Semantic Web рассказывают как о компоненте грядущей версии Web – так называемом Web 3.0. Каким на самом деле будет Web 3.0, мы можем только предполагать, но очевидно, что одним из главных его компонентов будет Semantic Web, в котором каждая Web-страница предоставляет также онтологию своего содержимого.

ПРИМЕЧАНИЕ

Термин «Web 3.0» является производным от «Web 2.0». Этим термином обозначают текущее состояние Web, которое принципиально отличается от того, в котором Web находился при его зарождении. С самого начала Web задумывался как распределенное по всему миру хранилище гипертекстовых документов, таковым Web остается и сегодня. Но с начала нынешнего века, с ростом числа пользователей, Web превратился в социальный феномен. Сегодня популярными сервисами являются не только те, которые предоставляют информацию, но и те, которые просто обеспечивают общение пользователей друг с другом. Такие сервисы получили название «социальные сети». Социальная сеть – совершенно новый феномен, который отличается от сети обычных гипертекстовых документов, как по принципам использования, так и по идеологии. Web социальных сетей – это Web 2.0. Одним из наиболее существенных отличий Web 2.0 от Web предыдущего поколения является то, что содержимое Web-страниц формируется пользователями, тогда как в старом Web это была задача разработчиков хранилищ документов – сайтов, на которых находились Web-страницы.

Формальная спецификация содержимого Web-документа дает возможность поисковой программе делать выводы о соответствии поискового запроса данному Web-документу не только на основе синтаксической информации, получаемой из текста этого документа, но и основываясь на семантике содержания данного документа. Это может кардинально улучшить качество Web-поиска, так как описание мира Web-страницы, понятное поисковой программе, дает последней гораздо больше информации, чем она может получить из неструктурированного текста.

Идеи умного Web давно были восприняты сообществом W3, в результате чего уже на протяжении более десяти лет ведутся работы по воплощению этих идей в жизнь. Первой задачей, которую необходимо решить для этого, является разработка стандартного языка, который был бы понятен всем поисковым программам. На настоящий момент разработаны два таких языка:

- Язык Resource Description Framework (RDF) – система описания ресурсов Web.
- Web Ontology Language (OWL) – язык онтологии Web. OWL можно рассматривать как расширение языка RDF.
- В следующих подразделах будут представлены описания этих языков.

Язык RDF

Язык RDF [6] разработан для того, чтобы описывать содержимое Web. В Semantic Web, когда говорят о каких-то сущностях Web, называют эти сущности ресурсами. RDF представляет собой язык для описания таких ресурсов. Ввиду того что описания семантики документов должны быть понятны компьютерам, необходимо разработать специальные программы-агенты, которые производили бы такое чтение. Также необходимо обеспечить возможность обмена информацией между различными программными агентами. Таким образом, под RDF подразумевается не только сам язык, но также и различные дополнительные программные модули, необходимые для обеспечения полноценного чтения и обмена информацией, записанной на этом языке. Этот факт подчеркивается в названии языка RDF.

Главный элемент языка RDF–это тройка, или триплет. Тройка представляет собой совокупность трех сущностей:

1. Субъект.
2. Объект.
3. Предикат.

Предикаты еще часто называют свойствами. Тройка имеет также представление в виде графа вида субъект–предикат–объект, где субъект и объект представлены как узлы, а предикат выступает в роли ребра, которое эти узлы соединяет.

С математической точки зрения, тройка представляет собой экземпляр некоторого бинарного отношения. Отношение – это множество последовательностей, состоящих в точности из n элементов для некоторого заранее определенного натурального числа n. Если n = 2, то отношение называется бинарным, т.е. бинарное отношение представляет собой множество пар. Например, отношение «семейные пары» задает множество пар элементов вида («муж», «жена»). Каждая тройка определяет одну пару из некоторого бинарного отношения, но кроме этого дополнительно задает еще имя отношения, т.е. если имеется пара («муж», «жена») отношения «семейные пары», то эту пару можно выразить тройкой («муж», «семейные пары», «жена»). Для полноценного описания, кроме задания отношений, необходимо еще задать ограничения на их содержимое. Например, для отношения «семейные пары» необходимо уточнить, что первый элемент каждой пары этого отношения должен быть мужчиной, а второй – женщиной. Для задания такого ограничения необходимо ввести понятия «мужчина» и «женщина», после чего задать ограничение, выражающее тот факт, что если имеется тройка вида («имя 1», «семейные пары», «имя 2»), то сущность «имя 1» должна быть экземпляром понятия «мужчина», а сущность «имя 2» – экземпляром понятия «женщина». Это делается посредством т.н. *высказываний*. Для высказываний существует свой язык, включающий переменные и логические операции. В качестве логических операций могут выступать: логическое «или» (дизъюнкция), логическое «и» (конъюнкция) и логическое следование (импликация). Имеются также кванторы существования и всеобщности, позволяющие ограничивать область применения высказывания. Язык RDF основан на математическом аппарате дескрипционной логики [7].

Дескрипционная логика (Description Logic – DL) базируется на формализмах семантических сетей [8] и фреймов [9], но использует аппарат математической логики. В математической логике производится явное разделение на синтаксис и семантику. Синтаксис задает язык, с помощью которого записываются различные высказывания об элементах мира данной логической системы. Семантика задает ту часть описываемого мира, которая удовлетворяет заданным ограничениям. Таких частей может быть более одной или даже бесконечно много. Каждая такая часть мира называется *моделью* данной логической системы. Опишем ограничения, налагаемые на синтаксис и семантику дескрипционных логик.

Синтаксис

Язык любой дескрипционной логики состоит из следующих элементов:

- Множество унарных предикатных символов, обозначающих имена понятий.
- Множество бинарных предикатных символов, обозначающих имена ролей.
- Рекурсивное определение термов понятий, задаваемое с помощью конструкторов на основе понятий и ролей.
- Понятия обозначают множества сущностей, которые им принадлежат, т.е. это классы в программистской терминологии. Роли задают отношения между понятиями. В качестве конструкторов термов выступают как операции логики первого порядка, такие, как перечисленные выше конъюнкция, дизъюнкция, ограничения универсальности и существования и т.д., так и операции, задающие ограничения ролей, т.е. бинарных отношений.

Ввиду того, что RDF предполагается использовать для описания ресурсов, распределенных по разным участкам Web, необходимо как-то решить проблему

идентификации имен узлов и ребер RDF графа, т.е. элементов троек. Для этого используется стандартный подход: каждый элемент описывается посредством так называемого Унифицированного Идентификатора Ресурса (URI – Uniform Resource Identifer [10]). Обычно URI представляет собой либо URL (Унифицированный Указатель Ресурса–Uniform Resource Locator [11]), содержащий информацию о местонахождении данного ресурса в Web, либо URN (Унифицированное Имя Ресурса – Uniform Resource Name [12]), позволяющий идентифицировать данный ресурс в некотором пространстве имен. Пространство имен представляет собой просто именованное множество элементов и используется, чтобы обеспечить уникальность имен этих элементов в Web.

В Semantic Web используются три стандартных пространства имен:

- rdf. В этом пространстве имен задаются имена, которые используются в RDF. URI для этого имени: http://www.w3.org/1999/02/22-rdf-syntax-ns#.
- rdfs. Здесь задаются имена, используемые в RDF Schema. Его URI: http://www.w3.org/2000/01/rdf-schema#.
- owl. Описываются имена, используемые в OWL. URI для этого имени: http://www.w3.org/2002/07/owl#.
- Для описания троек в языке RDF чаще всего используются две нотации:
- Нотация N-Triple [13]. В данной нотации каждый элемент тройки представляется посредством заключенного в угловые скобки URI. Элементы идут в порядке «субъект, предикат, объект» и заканчиваются точкой. Примеры этой нотации приведены ниже.
- Нотация, основанная на языке XML. Так как консорциум Web принял язык XML в качестве основного языка, то эта нотация использует наиболее часто.

Семантика

В математической логике в качестве модели логической системы обычно используется некоторое множество, назовем его M. Каждому элементу множества унарных предикатных символов дается в соответствие некоторое унарное отношение на этом множестве, а каждому элементу множества бинарных предикатных символов – бинарное отношение. Иначе говоря, каждому имени понятия соответствует некий класс-подмножество множества M, а каждому имени роли – бинарное отношение на множестве M. Если задаются какие-то ограничения на отношения, то они должны выполняться на всех отношениях. Каждое такое задание соответствия между именами и отношениями на множестве M (обозначим его через I) будем называть моделью данной дескрипционной логики, или ее интерпретацией. Интерпретация логической системы в некотором множестве представляет собой классический подход, но в RDF используется также интерпретация на графе. Иначе говоря, в качестве моделей может выступать граф, представляющий собой описанные выше тройки. Подробности об интерпретациях на графах можно узнать из документа [14].

В дескрипционных логиках проводится различие между т.н. терминологическим компонентом – TBox (terminological box) и компонентом суждений – ABox (assertional box). TBox содержит высказывания, касающиеся иерархии понятий, т.е. задающие отношения между понятиями, а ABox содержит высказывания, характеризующие отношения индивидов и понятий. Например, высказывание «каждый пользователь – это человек» задает отношение между понятиями «пользователь» и «человек», следовательно, принадлежит множеству TBox. Высказывание «Иван является пользователем» задает отношение между индивидом «Иван» и понятием «человек» и принадлежит множеству ABox. В дескрипционной логике элементы множества TBox представляют собой ограничения, задаваемые исключительно унарными предикатами (понятиями), в этом состоит их отличие от высказываний множества ABox. Различение высказываний на TBox

и ABox полезно, если рассматривать возможность построения процедуры логического вывода на моделях дескрипционных логик. Высказывания из TBox задают свойства «классификации», а высказывания из ABox – свойства, которые можно условно назвать свойствами «проверки экземпляра». Логический вывод по этим множествам может существенно различаться по производительности, поэтому имеет смысл реализовать отдельные алгоритмы вывода для каждого компонента.

Имеется множество стандартных видов дескрипционной логики, задаваемых различными ограничениями на виды отношений, которые в этих видах логики могут быть заданы. Здесь мы их перечислять не будем, обратимся лучше к конкретному примеру.

Пример спецификации онтологии на RDF

Предположим, что имеется некая база данных, содержащая информацию о пользователях. Информация о пользователе содержит следующие поля:

- Идентификатор (id).
- Имя (fname).
- Фамилия (sname).
- Возраст (age).
- Пол (sex).

База данных содержит информацию о трех пользователях:

Идентификатор	Имя	Фамилия	Возраст	Пол
1	Иван	Брусенко	39	мужской
2	Шамиль	Галимов	37	мужской
3	Татьяна	Волкова	23	женский

Таблица 1. Информация о пользователях.

Рассмотрим, как эти данные можно описать в виде онтологии на языке RDF. Принцип здесь простой: каждая строка таблицы представляется как экземпляр класса user. Элементы соответствующих столбцов в этом случае выступают как поля класса, что выражается в языке RDF в виде т.н. *свойств* – экземпляров класса rdf:Property. Чтобы выразить на языке RDF этот факт, будет использоваться специальный предикат rdf:type. Итак, имеем объявления полей как специальных свойств:

```
(usr:id, rdf:type, rdf:Property)
(usr:fname, rdf:type, rdf:Property)
(usr:sname, rdf:type, rdf:Property)
(usr:age, rdf:type, rdf:Property)
(usr:sex, rdf:type, rdf:Property)
```

ПРИМЕЧАНИЕ

Обратите внимание, что все имена полей объявлены в собственном пространстве имен usr.

Рассмотрим теперь, как задается таблица данных. Сначала для каждой строки таблицы зададим экземпляр класса «user»:

```
(usr:user1, rdf:type, usr:user)
(usr:user2, rdf:type, usr:user)
(usr:user3, rdf:type, usr:user)
```

Наконец, можно приступить к описанию самих данных:

```
(usr:user1,usr:id,1)
(usr:user1,usr:fname,Иван)
(usr:user1,usr:sname,Брусенко)
(usr:user1,usr:age,39)
(usr:user1,usr:sex,Мужской)
(usr:user2,usr:id,2)
(usr:user2,usr:fname,Шамиль)
(usr:user2,usr:sname,Галимов)
(usr:user2,usr:age,37)
(usr:user2,usr:sex,Мужской)
(usr:user3,usr:id,3)
(usr:user3,usr:fname,Татьяна)
(usr:user3,usr:sname,Волкова)
(usr:user3,usr:age,23)
(usr:user3,usr:sex,Женский)
```

Схема RDF

Схема RDF (RDF Schema, RDFS [15]) представляет собой расширение языка RDF, позволяющее описывать простые онтологии данных, находящихся в хранилищах RDF. Так же, как схема базы данных описывает структуру базы данных в виде заголовков таблиц и связей между ними, схема RDF позволяет описывать структуру RDF-хранилища. Структура описывает хранилище в терминах типов и отношений между ними. На самом деле, как в этом чуть позже убедится читатель, схема RDF позволяет описывать только классификации с некоторыми дополнительными отношениями. Чтобы описать более сложные виды отношений, необходимо привлекать более мощные средства, такие, как OWL.

В RDFS можно задавать классы, которые определяются в дескриптивной логике как унарные отношения. Для этого в RDFS определен специальный объект rdfs:Class – класс всех классов. Вообще, каждый объект RDF – это экземпляр класса rdfs:Resource, и rdfs:Class здесь не исключение. Но, с другой стороны, rdfs:Resource – это класс, а значит должен быть определен как экземпляр объекта rdfs:Class. Таким образом, объекты rdfs:Resource, и rdfs:Class определяются рекурсивно посредством друг друга – случай нередкий в языках описания онтологий.

Как уже говорилось выше, сказать, что данный объект является экземпляром данного класса, можно с помощью rdf:type. Например,

```
usr:user rdf:type rdfs:Class .
```

ПРИМЕЧАНИЕ

Выше мы обозначали тройки непосредственно, заключая их в скобки. Здесь же используется нотация N-Triple, в которой тройка записывается без скобок, но в конце записи ставится точка.

В RDFS также существует класс всех свойств, обозначаемый как rdf:Property. Все свойства являются экземплярами этого класса, а сам он, в свою очередь, является экземпляром класса rdfs:Class. Класс rdf:Property использовался в примере выше.

Для того чтобы сказать, что значения некоторого свойства являются экземплярами некоторого класса, т.е. чтобы задать типы свойств, используется свойство rdfs:range. Выражение:

```
P rdfs:range C
```

означает, что P – это экземпляр класса rdf:Property, а C – экземпляр класса rdfs:Class, и что все ресурсы, входящие в качестве объектов в тройки, предикат которых – это свойство P, являются экземплярами класса C. В ООП эта ситуация соответствует объявлению типа свойства. Например, объявление поля usr:age на языке C++ выглядит следующим образом:

```
int age;
```

задает тип int свойства age, т.е. говорит о том, что все объекты свойства age являются экземплярами класса int. Если rdfs:range позволяет задать тип значений, которые будет принимать некоторое свойство, то свойство rdfs:domain позволяет задать класс, чьим атрибутом является данное свойство. Выражение

```
P rdfs:domain C
```

говорит о том, что P – это экземпляр класса rdf:Property, а C – экземпляр класса rdfs:Class, и что все ресурсы, входящие в качестве субъектов в тройки, предикат которых – это свойство P, являются экземплярами класса C. Если свойство usr:age является атрибутом класса usr:user, то об этом можно сказать так:

```
usr:age rdfs:domain usr:user .
```

Итак, выражение языка C++:

```
namespace usr
{
  class user
  {
    int age;
  };
};
```

можно записать на RDFS следующим образом:

```
usr:age rdfs:range int .
usr:age rdfs:domain usr:user .
```

Свойство rdfs:subClassOf представляет собой аналог наследования в ООП. Выражение

```
C1 rdfs:subClassOf C2
```

означает, что тип C1 является подтипом типа C2, т.е. что каждый экземпляр класса C1 является также и экземпляром класса C2.

В RDFS также определены элементы, позволяющие задавать контейнеры (классы, унаследованные от rdfs:Container) и коллекции (класс rdf:List). Для подробного

ознакомления со всеми элементами словаря RDF читатель может обратиться к официальному описанию RDFS на сайте консорциума W3 [15].

Язык OWL

OWL (Web Ontology Language [17]) представляет собой язык, предназначенный для описания онтологий и разработанный консорциумом W3 специально для этих целей. OWL построен как расширение RDF и RDFS. Это означает, что основная конструкция – это тройка языка RDF. В этом контексте язык OWL можно рассматривать как расширенный вариант RDFS, позволяющий не только описывать классы и свойства, но также задавать ограничения на их использование. На языке дескрипционной логики это означает, что логика, лежащая в основе OWL, содержит кроме описания отношений также и аксиомы, задающие соотношения между данными отношениями и различного рода ограничения последних.

Базовым элементом языка OWL является класс всех классов, определяемый как owl:Class. Класс owl:Class – это экземпляр рассмотренного выше класса rdfs:Class. Любой OWL-класс должен быть задан как экземпляр класса owl:Class. Например, если мы хотим определить класс Human (человек), то должны задать тройку

```
Human rdf:type owl:Class
```

В языке OWL также присутствуют два предопределенных класса:

- Класс owl:Thing (сущность), который обозначает множество всех индивидов.
- Класс owl:Nothing (ничто), обозначающий пустое множество.

Каждый класс OWL является дочерним классом класса owl:Thing и родительским классом класса owl:Nothing. Наследование классов в языке OWL задается с помощью конструкции rdfs:subClassOf, т.е. точно так же, как и в языке RDF Schema.

В OWL существует разделение свойств на два класса:

- Объектные свойства используются для связывания индивидов друг с другом. Объектные свойства – это экземпляры класса owl:ObjectProperty.
- Свойства типов данных связывают индивидов с так называемыми значениями типов данных (data values). Под значениями здесь подразумеваются RDF-литералы, или типы данных, определенные в XML Schema. Свойства типов данных – это экземпляры класса owl:DatatypeProperty.

Классы owl:ObjectProperty и owl:DatatypeProperty являются дочерними классами класса rdf:Property.

Язык OWL позволяет описывать различные характеристики классов и свойств, которые обычно задаются как разного рода ограничения на структуру связей между своими экземплярами. Эти ограничения выражаются в виде предопределенных соотношений, называемых в языке OWL аксиомами. В этом состоит основное отличие языка OWL от RDFS. Эти ограничения позволяют выражать в онтологии более тонкие вещи, чем с помощью RDFS. Например, в языке OWL имеется аксиома owl:equivalentClass, которая позволяет сказать, что множества экземпляров двух данных классов совпадают. Например, можно задать класс с именем ГенеральныйДиректор, а можно еще сказать о классе ГлаваКомпании. Тогда можно определить эквивалентность этих классов следующим образом:

```
ГенеральныйДиректор owl:equivalentClass ГлаваКомпании .
```

Также можно задавать ограничения для атрибутов (свойств) классов. Например, ограничения количества элементов (cardinality constraints) накладывают ограничения на мощность множества значений данного свойства, если в качестве субъекта выступает некоторый конкретный класс. Так, можно указать, что количество значений свойства Игрок класса ФутбольнаяКоманда должно равняться числу 11.

Таких ограничений в языке OWL множество. Но все они подобраны таким образом, чтобы не снизить производительность алгоритма логического вывода по фактам, которые описаны в онтологии. Более подробную информацию о языке OWL читатель может почерпнуть из [16].

Синтаксис XML удобен для чтения машинами, но не слишком удобен для человека. Поэтому обычно документы с онтологиями, написанными на языке OWL, не редактируют непосредственно, а используют для редактирования специализированные программы. Одна из таких программ называется Protégé [18]. Эта программа бесплатна для использования и может быть загружена с сайта проекта Protégé [18]. OWL-код, записанный в файле онтологии, представляется непосредственно в виде коллекций классов и их свойств, а также в виде различных их ограничений, как это описано выше. Программа Protégé предоставляет возможность задавать запросы на языке SPARQL [19] к открытой онтологии, а также производить различные логические манипуляции над онтологией.

Кроме программ редактирования онтологий имеется большое количество реализаций Web-сервисов онтологий – хранилищ библиотек онтологий. Такие хранилища подобны серверам баз данных, предоставляемых разработчиками сервиса программистам, которые наполняют содержимое этих сервисов конкретными данными. Существуют как коммерческие, так и бесплатные реализации таких сервисов. Один из наиболее популярных сервисов такого рода называется OWLIM [20], написан на Java и, как декларируется его разработчиками, представляет собой быстрейшую на сегодняшний день реализацию хранилища. Более подробный список реализаций RDF-хранилищ приведен в [1].

Заключение

В данной работе сделана попытка в популярной форме познакомить читателя с понятием онтологии и приложениями этого понятия в компьютерных системах. Чтобы изложение не было слишком абстрактным, было приведено описание одного конкретного примера использования онтологий в Интернет – описания содержимого Web-страниц в виде онтологий. Для таких описаний консорциумом W3 разработаны специальные языки: RDF и OWL. В статье было приведено их краткое изложение.

К сожалению, ограничения объема, налагаемые на журнальную статью, не дают возможности рассказать о понятии «онтология» подробно. Заинтересованный читатель может получить подробную информацию в книге «Онтологии в компьютерных системах» [1]. В этой книге рассмотрены различные аспекты применения онтологий в информационных системах, и многие из этих применений не ограничиваются Интернет.
Автор надеется, что приведенная в работе информация будет полезна читателю не только в качестве «расширения кругозора», но и даст пищу для реализаций конкретных проектов в самых различных областях информатики.

Список литературы

1. Лапшин В.А. Онтологии в компьютерных системах. М.: Научный мир, 2010.
2. Gruber T.R. The role of common ontology in achieving sharable, reusable knowledge bases // Principles of Knowledge Representation and Reasoning. Proceedings of the

Second International Conference. J.A. Allen, R. Fikes, E. Sandewell – eds. Morgan Kaufmann, 1991, 601-602.
3. Бениаминов Е.М. Алгебраические методы в теории баз данных и представлении знаний. М.: Научный мир, 2003.
4. Guarino N. Formal ontology in information systems // Proceedings of FOIS'98, Trento, Italy, 6-8 June 1998. Amsterdam, IOS Press, 1998. 3-15.
5. Когаловский М., Калиниченко Л. Концептуальное моделирование и онтологические модели // Онтологическое моделирование. Труды симпозиума в г. Звенигороде, 19-20 мая, 2008.
6. Описание концепций языка RDF на сайте W3. http://www.w3.org/TR/rdf-concepts/.
7. The description logic handbook: Theory, implementation, and applications. F. Baader, D. Calvanese, D. McGuiness, D. Nardi, P. Patel-Schneider. Cambridge: University Press, 2003.
8. Quillian M. Word concepts: A theory and simulation of some basic capabilities // Behavioral Science, 1967. 12. 410-430.
9. Minsky M. A framework for representing knowledge. J. Haugeland – ed. Mind Design. The MIT Press, 1981.
10. Uniform resource identifier (URI): Generic Syntax. http://tools.ietf.org/html/rfc3986.
11. Uniform resource locators (URL). http://tools.ietf.org/html/rfc1738.
12. URN syntax. http://tools.ietf.org/html/rfc2141.
13. Спецификация нотации N-Triple на сайте консорциума W3. http://www.w3.org/2001/sw/RDFCore/ntriples/.
14. RDF Semantisc. http://www.w3.org/TR/2004/REC-rdf-mt-20040210/.
15. RDF Vocabulary Description Language 1.0: RDF Schema. http://www.w3.org/TR/rdf-schema/.
16. Сайт консорциума World Wide Web. http://www.w3.org/.
17. Описание языка OWL на сайте W3. http://www.w3.org/TR/owl-ref/.
18. Сайт редактора онтологий OWL Protege. http://protege.stanford.edu/.
19. SPARQL query results XML Format. http://www.w3.org/TR/rdf-sparql-XMLres/.
20. Сайт OWLIM Semantic Repository. http://www.ontotext.com/owlim/.

Онтологии для реализации обратной трассировки при разработке и сопровождении программ

Ontology's for realization of reverse tracing at development and maintenance of programs

Статья опубликована в выпуске журнала № 4 за 2009 год. [16.12.2009] «Программные продукты и системы». Международный журнал.

Аннотация: *Дается краткая характеристика метода обратной трассировки кода, направленного на улучшение понимаемости программ в процессе их разработки и сопровождения. Обосновывается применение онтологий как средства представления знаний в процессе разработки и сопровождения программ. Предлагается вариант онтологии, предназначенной для представления знаний в процессе отслеживания и устранения ошибок программного проекта.*

Abstract:*The short characteristic of a method of reverse trace of the code directed on improvement of intelligibility of programs in the course of their working out and support is given. Application ontology as means of representation of knowledge in the course of development and maintenance of programs is proved. The variant ontology, intended for representation of knowledge in the course of tracing and elimination of errors of the program project is offered*

авторы: Рогальчук В.В., Хомоненко А.Д.

Авторы: http://www.swsys.ru/index.php?page=infou&id=5705 - Петербургский государственный университет путей сообщения , http://www.swsys.ru/index.php?page=infou&id=5706 - Петербургский государственный университет путей сообщения Доктор технический наук Ph.D.
Ключевые слова: сопровождение программ, разработка программ, программное обеспечение, представление знаний, ошибки, отношения, онтологии, обратная трассировка программ

Keywords: maintenance of programs, development of programs, the software, representation of knowledge, errors, relations, ontology, reverse trace of programs

На эффективность процессов разработки и сопровождения программного обеспечения (ПО) большое влияние оказывает используемая технология. Одним из подходов, способствующих повышению эффективности разработки и сопровождения ПО, является метод обратной трассировки [1]. Названный метод заключается в предоставлении разработчику инструментария, обеспечивающего поиск и навигацию от элементов программного кода к связанным с его созданием и изменением в ходе развития проекта документам. Это способствует лучшему пониманию и ускорению изучения кода программ. Так как задача изучения программ является одной из самых трудоемких в процессах разработки и сопровождения [2], использование обратной трассировки позволяет снизить общую стоимость проекта разработки. Наличие дополнительного инструментария увеличивает эффективность поиска и устранения ошибок, за счет чего повышается надежность программной системы.

При реализации метода обратной трассировки кода программ требуется решить задачу представления знаний о разрабатываемом проекте ПО. Одним из наиболее перспективных подходов к представлению знаний, по мнению многочисленных специалистов, является использование онтологий [3].

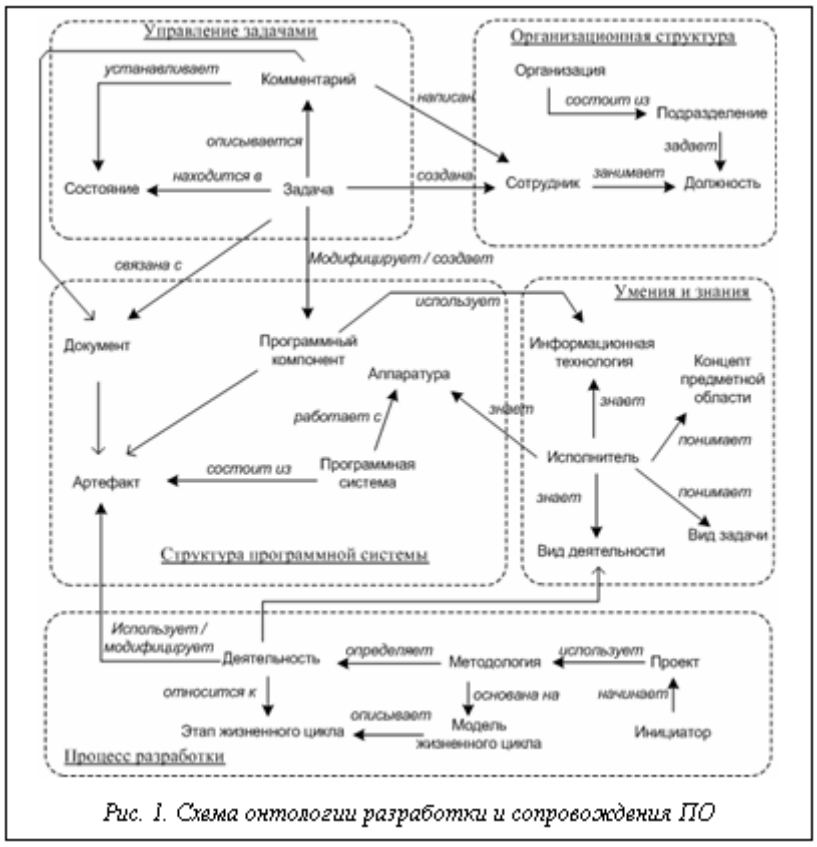

Рис. 1. Схема онтологии разработки и сопровождения ПО

Онтология – модель представления знаний, комбинирующая в себе свойства семантических сетей и логических моделей. Понимание термина «онтология» варьируется в зависимости от области применения, но среди множества различных определений чаще всего используется определение Грубера: онтология – точная спецификация концептуализации [3]. Под концептуализацией понимается обобщение понятий и информации, необходимых для описания объектов и решения задач в выбранной области знаний – свойства, отношения, аксиомы, утверждения, ограничения.

В [3] предложено рассматривать онтологию как базу знаний, которая может отчуждаться от разработчика и физически делиться между пользователями. Онтологии

как средство представления знаний разрабатываются для большого количества предметных областей.

Систематическое описание онтологий для области программной инженерии приводится в [4]. В частности, авторы описывают онтологии для методологий разработки ПО, его сопровождения и измерения характеристик. Для сопровождения ПО предложена следующая группа подонтологий: описания элементов программной системы, используемых в создании компьютерных знаний, организационной структуры, процесса сопровождения и домена приложения. Как показывает проведенный анализ, данные онтологии не в полной мере поддерживают решение задачи обеспечения понимаемости кода при обратной трассировке программы. В состав реализующей обратную трассировку среды разработки должны входить средства интеграции с существующими системами управления задачами и отслеживания сообщений об ошибках. Следовательно, в состав онтологической системы необходимо включить подонтологию описания задач разработки и сопровождения ПО, а также подонтологию описания ошибок, обнаруживаемых в процессе инспектирования программного проекта. На рисунке 1 показаны основные элементы пяти подонтологий и некоторые из связей между ними.

Подонтология управления задачами для разработчиков создана для интеграции системы управления задачами с системой реализации обратной трассировки. Любые реализации систем управления задачами оперируют понятиями Задача и Комментарий. Задача обсуждается Сотрудниками организации-разработчика. В ходе обсуждения с Задачей связывается множество Комментариев, каждый из которых может устанавливать новое Состояние задачи (Новая задача, Изучается, Отложена, Выполнена, Закрыта, Возобновлена и т.д.). В составе онтологической системы каждый комментарий является Документом, к которому могут быть проведены трассировочные связи от Артефактов, используемых и изменяемых в ходе решения задачи. Задачи могут ссылаться на другие задачи через отношения иерархии и взаимозависимости, а также на экземпляры других классов в базе знаний.

Онтология описания ошибок

В процессах разработки и сопровождения ПО постановка новых задач часто связана с поступлением информации об обнаруженных ошибках. Предлагаемая в настоящей статье подонтология разработана для интеграции Bug-tracking-систем (систем отслеживания ошибок) в трассировочную среду разработки и содержит основные понятия, используемые в этих системах (рис. 2). На рисунке 2 классы, представленные в онтологии на рисунке 1, выделены подчеркиванием.

Описание Ошибки содержит время и описание ее обнаружения (элементарные поля свойств «дата обнаружения», «описание»). Указывается Сотрудник, создавший описание. Оценка серьезности ошибки связывается с Уровнем серьезности, также указывается вынесший оценку Сотрудник. Информация об оценках серьезности в дальнейшем может использоваться для ранжирования задач.

Для реализации онтологии естественно воспользоваться программным инструментарием, предназначенным для проектирования, редактирования и анализа онтологий. Из множества современных инструментальных сред поддержки разработки онтологий (Ontolingua, WebOnto, Protege, OntoSaurus и др.) наибольшее распространение получил редактор Protege.

Пример реализации онтологии в среде Protege

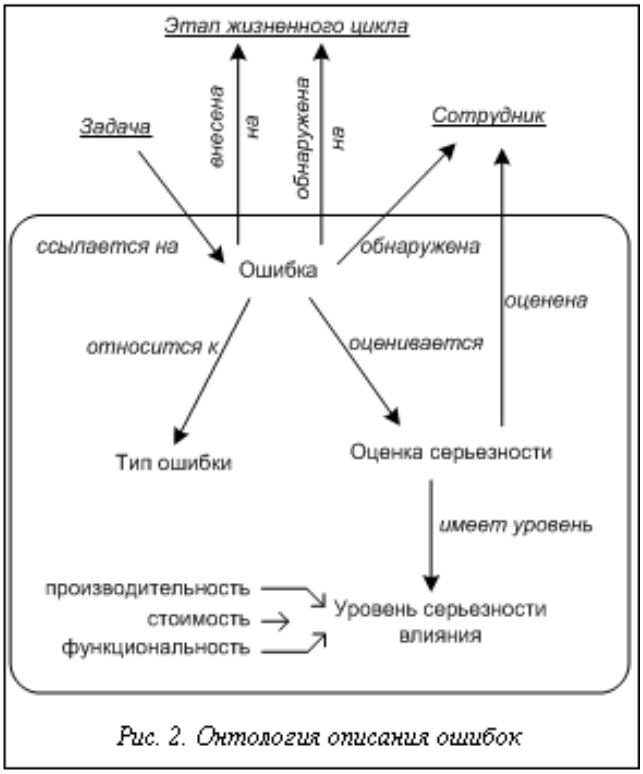

Рис. 2. Онтология описания ошибок

Программа Protege предназначена для построения (создания, редактирования и просмотра) онтологий прикладной области. Ее первоначальная цель – помощь разработчикам ПО в создании и поддержке явных моделей предметной области и включение этих моделей непосредственно в программный код. Protege позволяет проектировать онтологии, разворачивая иерархическую структуру абстрактных или конкретных классов и слотов. Редактор Protege основан на фреймовой модели представления знания OKBC (Open Knowledge Base Connectivity) и снабжен рядом плагинов, что позволяет адаптировать его для редактирования моделей, хранимых в разных форматах (стандартный текстовый, в БД JDBC, UML, языков XML, XOL, SHOE, RDF и RDFS, DAML+OIL, OWL).

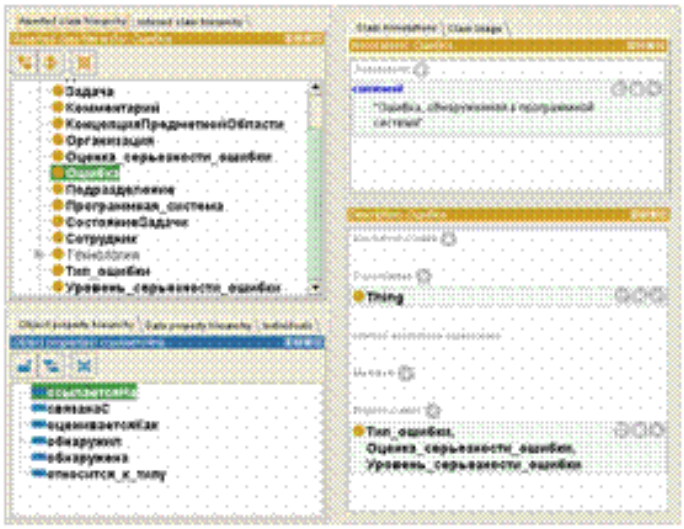

Рис. 3. Вид среды редактора Protege в процессе создания онтологии

Структура онтологии аналогична иерархической структуре каталога. На основе сформированной онтологии Protege позволяет генерировать формы получения знаний для введения экземпляров классов и подклассов. Рассмотрим реализацию разработанной авторами онтологии в среде Protege (рис. 3). На вкладке Asserted class hierarchy содержится иерархия классов общей онтологии процесса разработки ПО с выбранным классом Ошибка.

На вкладке Object properties для выбранного класса Ошибка указываются свойства, в которых этот класс участвует. На вкладке Annotations приводится комментарий для класса, а на вкладке Description дается описание его свойств: наследуемые суперклассы, логически выведенные суперклассы, хранимые в онтологии объекты этого класса и несовместные классы.

Рассмотрим пример использования предложенной онтологии описания ошибок для проведения экспертизы программного продукта, описываемой в [5]. Автором предлагается периодическое проведение экспертизы программного кода для повышения эффективности процесса поиска и устранения ошибок. При этом для выявления причин отклонения производительности и определения необходимых корректирующих или предупредительных действий предлагаются правила анализа экспертизы, проводится анализ количества обнаруженных ошибок, и, если их число отклоняется от нормального значения, в соответствии с вероятной причиной выполняется определенный набор действий.

Предлагаемая в статье онтология применяется для хранения протоколов инспекции. Перед использованием подсистемы производится предварительное заполнение базы знаний объектами классов, участвующих в описании протоколов (в частности, следует внести информацию о работниках организации, описать используемую модель жизненного цикла и виды деятельности, внести типы ошибок, шкалу уровней серьезности ошибок). Далее в базу знаний из протоколов инспекции вносится информация об обнаруженных ошибках. Для каждого описания ошибки создаются экземпляры, принадлежащие соответствующим классам, устанавливаются связи между ними, вносится информация о связанных с ошибками задачах. Изменения, сделанные в ходе работы над программным кодом, отслеживаются с помощью среды разработки и репозитория программного кода, и в базу знаний вносится информация о взаимосвязи элементов программного кода и исправленных ошибок. Всю сохраненную информацию можно взять для описанного выше финального анализа результатов инспекции.

Предложенная онтология может использоваться для представления знаний о процессе поиска и устранения ошибок, а также для интеграции с различными реализациями систем отслеживания и устранения ошибок.

Литература

1. Рогальчук В.В., Хомоненко А.Д. Метод обратной трассировки и оценивание его влияния на стоимость разработки программного обеспечения // Научно-технические ведомости СПбГПУ. СПб, 2008. № 4 (62). С. 146–151. (Информатика. Телекоммуникации. Управление).
2. Grubb P., Takang A.A. Software maintenance: concepts and practice (2nd edition). Singapore: World Scientific Publishing, 2003. 369 p.
3. Гаврилова Т.А., Хорошевский В.Ф. Базы знаний интеллектуальных систем. СПб: Питер, 2001. 384 с.
4. Calero C., Ruiz F., Piattini M. Ontologies for Software Engineering and Software Technology. Springer-Verlag Berlin Heidelberg. 2006. 339 p.
5. Джалота П. Управление программным проектом на практике. М.: Изд-во «ЛОРИ», 2005. 223 с.

Онтология анализа данных

Николай Некипелов, Акобир Шахиди

"Отчего же не вырвать? Вырвать можно. Только тут понимать надо, без понятия нельзя… Зубы разные бывают. Один рвешь щипцами, другой козьей ножкой, третий ключом… Кому как."
А.П.Чехов

Введение

Потоки текстовой и числовой информации ежедневно порождаются и оседают в хранилищах данных. Насколько полно на практике используются все те закономерности, которые кроются в этих данных и, возможно, представляют большую ценность? Можно предположить, что процент переработки "сырых" данных в практически значимые знания пока что весьма скромен. Даже богатый арсенал классической статистики используется далеко не полностью, не говоря уже о более современных методах нелинейного анализа. "Там, где обязаны поклоняться солнцу, законы теплоты будут слабо поняты" Речь о том, что в нашей стране, хотя статистика и не обзывалась "продажной девкой буржуазии", длительное время осуществлялось неприятие формальной статистики. Какая тут статистика, если сами данные должны были соответствовать идеологическим установкам государства. Ситуация усугубляется тем, что в последнее время активно развиваются новые методы анализа данных и извлечения знаний, базирующиеся на иных, нежели традиционная интегро–дифференциальная парадигма, подходах. Имеются в виду методы эволюционного моделирования и методы машинного обучения. Термин "эволюционное моделирование" в настоящее время является достаточно устоявшимся, и общепринято под этим термином подразумевать генетические алгоритмы и искусственные нейронные сети. Термин "машинное обучение" оставляет больше возможностей для дискуссий о том, какие методы имеются в виду, в частности, сюда относятся деревья решений.

Что такое онтология?

Как ориентироваться в этом многообразии инструментов? Какой из них выбрать для решения конкретной задачи? В сложившейся ситуации очень кстати приходится сравнительно новый термин – "онтология". Онтология – это точная спецификация некоторой предметной области. Она обеспечивает словарь для представления и обмена знаниями об этой предметной области и множество связей, установленных между терминами в этом словаре. В простейшем случае построение онтологии сводится к:

- Выделению концептов – базовых понятий данной предметной области;
- Построению связей между концептами – определению соотношений и взаимодействий базовых понятий.

Одним из преимуществ использования онтологий в качестве инструмента познания является системный подход к изучению предметной области. При этом достигаются:

- Систематичность – онтология представляет целостный взгляд на предметную область;
- Единообразность – материал, представленный в единой форме гораздо лучше воспринимается и воспроизводится;
- Научность – построение онтологии позволяет восстановить недостающие логические связи во всей их полноте.

Онтология анализа данных

Так как знания носят личностный характер, одну и ту же предметную область можно описать разными онтологиями. Особенно это касается плохо формализуемых предметных областей или при наличии большого числа спорных вопросов.

Математическая статистика

Для решения задач, связанных с анализом данных при наличии случайных и непредсказуемых воздействий, математиками и другими исследователями за последние двести лет был выработан мощный и гибкий арсенал методов, называемых в совокупности математической статистикой. За это время накоплен большой опыт успешного применения этих методов в разных сферах человеческой деятельности, от экономики до космических исследований. И при определенных условиях эти методы позволяют получать оптимальные решения. Например, одна из задач, решаемых в радиолокации – обнаружение известного сигнала на фоне аддитивной помехи в виде белого шума. Методы математической статистики решают эту задачу оптимальным образом и трудно себе представить необходимость применения других подходов к решению этой задачи. В тоже время, задача разрешения близко расположенных целей в условиях более сложной помеховой обстановки линейными статистическими методами решается менее успешно.

Эволюционное моделирование

На сегодняшний день, говоря об эволюционном моделировании, обычно имеют в виду генетические алгоритмы и искусственные нейронные сети. Термин "эволюционное моделирование" обязан своим происхождением источнику заимствования идей, лежащих в основе этой парадигмы. Если в основе классических подходов лежат формализованные каким-либо образом *знания* человека о предметной области, то для нейронной сети аналитическая форма представления знаний недоступна, все что она может – это запомнить и обобщить предъявленные ей на этапе обучения эмпирические зависимости между входными факторами и результирующими значениями. То есть нейронная сеть строит модель некоего процесса и в дальнейшем воспроизводит его поведение. Это дает повод некоторым исследователям утверждать, что искусственные нейросети моделируют свойственные человеку *приемы мышления*. По нашему мнению, для практического

использования нейросетевых технологий достаточно того обстоятельства, что нейросети в состоянии строить сложные нелинейные модели процессов, а как на самом деле устроены человеческие мозги – дело десятое. Важно другое – качество модели зависит от качества обучающих данных (тут все как у людей).

Генетические алгоритмы используют механизмы генетической эволюции, которые в общем виде могут быть сформулированы так: чем выше приспособленность особи, тем выше вероятность того, что в его потомстве эта приспособленность будет выражена еще сильнее. Трактовка процесса приспособления как оптимизационного процесса приводит к идее использования генетических алгоритмов при обучении нейронных сетей. Причем, если градиентные методы обучения гарантирую нахождение локального минимума, то генетический алгоритм обеспечивает глобальную оптимизацию.

Область применения

Методами эволюционного моделирования решается широкий класс задач: классификация образов, кластеризация, аппроксимация, прогноз данных, оптимизация, ассоциативная память, управление динамическими объектами. Причем в силу всего вышесказанного, нейронные сети в сравнении с методами математической статистики справляются с перечисленными задачами тем успешнее, чем хуже формализуема задача.

Достоинства нейросетей

- Одним из основных достоинств нейронных сетей является то, что они имеют широкую область применения. Деревья решений напротив, ограничены в рамках задач классификации, следует заметить, что существуют алгоритмы решающие задачи прогнозирования, но они значительно уступают нейронным сетям;
- Нейронные сети по своей природе являются универсальными аппроксиматорами и позволяют моделировать очень сложные закономерности, что, скажем, не доступно классическим регрессионным моделям;
- Нет необходимости заранее знать вид аппроксимируемой функции;
- Нейронная сеть может быть легко дообучена с учетом вновь поступивших данных, для деревьев решений на сегодняшний день это большая проблема, поскольку не разработана методика "достроения" дерева, приходится строить дерево с нуля, не учитывая ранее построенное;
- Существуют нейросетевые парадигмы, например, карты Кохонена, в которых процесс обучения происходит без учителя, т.е. сеть сама разбирает структуру данных;
- Другая нейросетевая парадигма РБФ – сети очень быстро обучаются, хотя надо заметить, что так называемое "проклятие размерности" касается их в большей степени.

Машинное обучение

Цель методов машинного обучения – получение простых классифицирующих выражений, которые были бы легко понятны для человека. Достоинством таких методов является то, что во время работы того или иного метода не требуется участие человека.

Область применения

В исследовании, проведенном в рамках европейского проекта StatLog, был проведен анализ статистических методов (дискриминантый анализ, кластер-анализ и т.д.), деревьев решений(C4.5, AC2, CART, NewID, CN2, Itrule и т.д.) и нейронных сетей (многослойные сети, РБФ-сети, карты Кохонена) для решения задач классификации. Данные были взяты из различных предметных областей: распознавание образов (рукописного текста, автомобилей), медицинская диагностика (диабет, травмы головы, сердечные заболевания), молекулярной биологии (распознавание структуры ДНК) выдача кредитов и т.д.

В ходе исследования выяснилось, что деревья решений показали наилучшие результаты в решении следующих задач:

1. Оценка кредитоспособности кандидата на получение кредита;
2. Диагностика неисправностей в технических системах;
3. Размещение радиаторов в Space Shuttle.

Достоинства деревьев решений

- На обучение деревьев решений требуется гораздо меньше времени, чем, например, на обучение нейронных сетей;
- Результат работы представляется в легко интерпретируемом для человека виде. Классификационная модель, представленная в виде дерева является интуитивно понятной для человека, в отличие от нейронных сетей, являющихся по своей природе черным ящиком;
- На вход алгоритма деревьев решений можно подавать любое количество параметров, алгоритм сам выберет наиболее значимые параметры и только они будут фигурировать в построенном дереве. Это избавляет пользователя от необходимости определять входные параметры. Опять же, при использовании нейронных сетей мы должны очень осторожно подходить к вопросу о входных полях, так, с ростом количества входных полей, увеличивается время затрачиваемое на процесс обучения, который и так является очень долгим и вызывает много нареканий;
- Точность прогноза деревьев решений сопоставима с другими методами построения классификационных моделей (статистические методы, нейронные сети);
- Существуют масштабируемые алгоритмы деревьев решений SLIQ, SPRINT, т.е. с ростом числа примеров время затрачиваемое на обучение растет линейно для построения деревьев решений на сверхбольших базах данных;
- Алгоритмы построения деревьев решений имеют методы специальной обработки пропущенных данных;
- Классические и современные методы статистики используемые в задачах классификации работают только с числовыми данными, деревья решений успешно работают как с числовыми так и строковыми значениями. Кроме того, некоторые из статистических методов являются параметрическими, т.е. мы заранее должны знать вид модели или зависимость между зависимыми и независимыми переменными. Например, классификаторы, построенные по принципу максимального правдоподобия, предполагают, что данные имеют нормальное распределение;
- Позволяют извлекать правила на естественном языке, например: Если **возраст > 35** И **доход > Среднего** То **Выдать кредит**.

Заключение

На нашем форуме иногда можно встретить довольно раздраженные реплики по поводу всех этих умностей. Особой нелюбовью почему то пользуются нейронные сети. Нам бы хотелось призвать этих авторов к большей сдержанности и сказать следующее.

Во-первых, если трезво посмотреть вокруг, выясняется, что при помощи нескольких магических слов, таких как *нейросеть, персептрон, факторный анализ, регрессионный анализ…* , нельзя решить всех нерешенных проблем. "Очень редко удается открыть одновременно несколько тайн природы одним и тем же ключом". (К. Шеннон).

Во-вторых, эффективность методик нелинейного оценивания (имеется в виду нейрокомпьютниг) может быть повышена при сочетании их с уже известными линейными статистическими методами. Пример – сети РБФ, в которых настройка весов скрытого слоя ведется с помощью генетического алгоритма, а веса выходного слоя рассчитываются старым добрым методом псевдообратных матриц.

Это всего лишь инструмент. Как им пользоваться, решает в конце концов человек. Кстати история, описанная Чеховым в рассказе "Хирургия" (откуда взят эпиграф), произошла только потому, что вместо доктора, уехавшего жениться, больных принимал фельдшер Курятин.

Литература

1. Т. Гаврилова "Онтология для изучения инженерии знаний", Труды Международной научно – практической конференции KDS-2001, 2001 г.
2. Г.К.Вороновский, К.В.Махотило, С.Н.Петрашев, С.А.Сергеев "Генетические алгоритмы, искусственные нейронные сети и проблемы виртуальной реальности", Харьков Основа 1997 г.
3. Гвидо Дебок, Тейво Кохонен "Анализ финансовых данных с помощью самоорганизующихся карт", 2001 г.
4. В.В.Круглов, В.В.Борисов "Искусственные нейронные сети", 2001 г.
5. Л.А.Сошникова, В.Н.Тамашевич, Г.Уебе, М.Шефер "Многомерный статистический анализ в экономике", 1999 г.
6. J. Ross Quinlan. C4.5: Programs for Machine learning. Morgan Kaufmann Publishers 1993.
7. Machine Learning, Neural and Statistical Classification. Editors D. Mitchie et.al. 1994.

Обзор применения онтологий в моделировании и управлении

Кудрявцев Д.В.

Введение

Онтология — формальная спецификация разделяемой концептуальной модели [Studer,1998], где

- под «концептуальной» моделью подразумевается абстрактная модель предметной области, описывающая систему понятий предметной области;
- под «разделяемой» подразумевается согласованное понимание концептуальной модели определенным сообществом (группой людей);
- «спецификация» подразумевает описание системы понятий в явном виде;
- «формальная» подразумевает, что концептуальная модель является машиночитаемой.

Онтология состоит из классов сущностей предметной области, свойств этих классов, связей между этими классами и утверждений, построенных из этих классов, их свойств и связей между ними.

В рамках [Эталонные модели, 2006] было проведено детальное исследование понятия «онтология» и особенно областей применения онтологий. В результате систематизация знаний в области онтологий, предложенная в [Гаврилова, 2003], была расширена (рис.1):

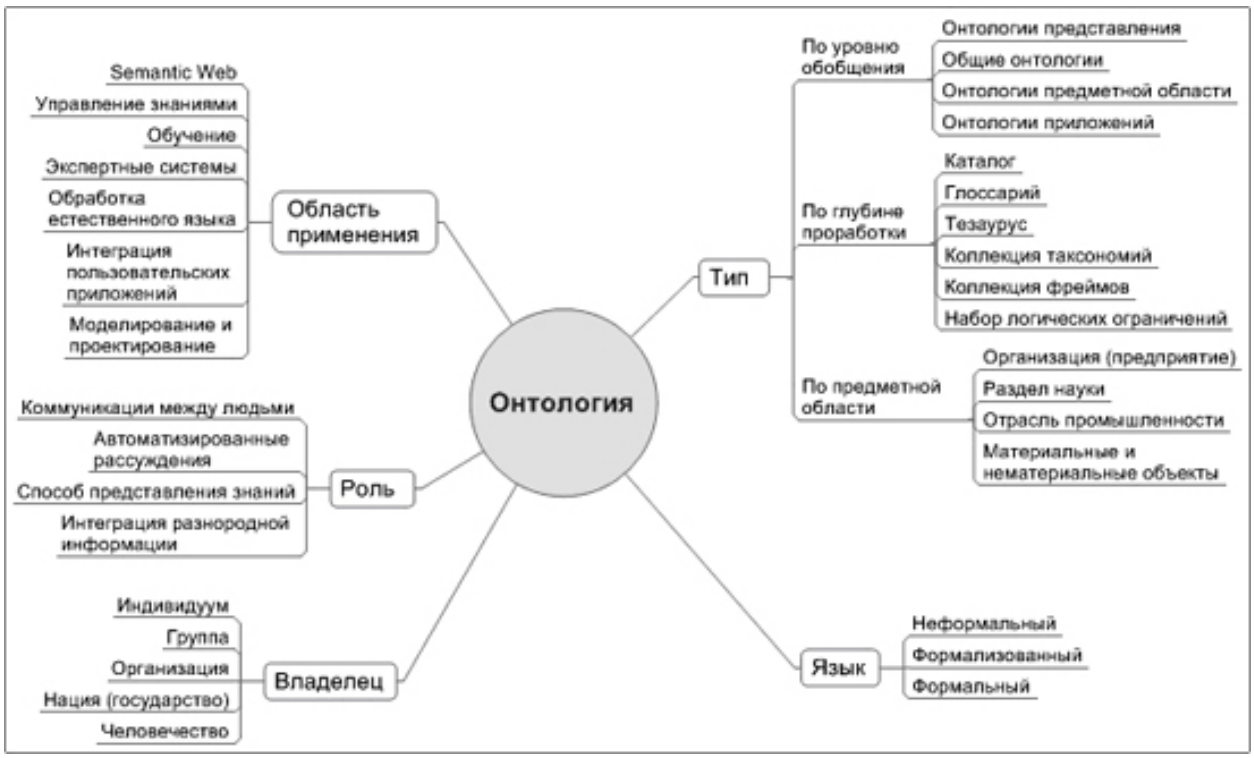

Рис. 1. Систематизация знаний в области онтологий

К настоящему моменту существует ряд примеров использования онтологий в организационном моделировании и управлении:

1. Enterprise Project [Stader, 1996; Uschold, 1997],
2. Process Specification Language (PSL) project [Bock, 2005],
3. Toronto Virtual Enterprise (TOVE) project [Fox, 1992; Gruninger, 2000],
4. SUPER Project [Born, 2007; Hepp, 2007],
5. Понятийное и объектное моделирование властных структур на региональном уровне ИПУСС РАН [Виттих и др., 2006]
6. Конфигурирование услуг электронного государства на основе онтологий —Onto-Gov [Abecker, 2004]

Далее приводится краткое описание указанных примеров.

Enterprise Project и Онтология Предприятия (Enterprise Ontology)
http://www.aiai.ed.ac.uk/project/enterprise/

Целью проекта Enterprise, выполненного совместно представителями университета Эденбурга, IBM, Unilever и Lloyd's Register [Uschold, 1997], являлось моделирование бизнес-среды для поддержки менеджеров в принятии взвешенных стратегических, тактических и операционных решений. В свою очередь основная роль моделирования в проекте Enterprise состоит в формировании комплексного взгляда на организацию. Для достижения, использования и поддержки такого комплексного взгляда на организацию требуются мощные средства интеграции, коммуникации, гибкости и поддержки.

В результате проекта Enterprise был разработан Enterprise Tool Set (ETS) для информационной поддержки предоставления комплексного взгляда на предприятие.

В качестве базовой модели поддержки моделирования предприятия используются модели процессов, которые обеспечивают процессно-ориентированный взгляд и которые могут быть реализованы в исполнительной (running) системе. Для разработки моделей процессов был создан Procedure Builder. В большинстве организаций существует множество используемых инструментов (приложений). Было решено поддерживать

интеграцию существующих инструментов с минимальными их изменениями, вместо того чтобы заменять существующие инструменты и их интерфейсы. Для реализации такого подхода был создан Agent Toolkit — архитектура, основанная на агентах (agent-based architecture), совмещенная с библиотекой, поддерживающей процесс добавления инструментов к системе.

Кроме поддержки интеграции инструментов, предложена поддержка реализации (исполнения) процессов. Task Manager обеспечивает как интеграцию инструментов, так и поддержку самих моделей процессов. Кроме того, он обеспечивает поддержку реализации процессов во времени (agenda-style).

Для достижения такой высокоуровневой интеграции и обеспечения эффективной коммуникации компонент, необходимо согласование используемых понятий. Для этого была разработана онтология предприятия (Enterprise Ontology).

В результате состав Enterprise Tool Set следующий (рис. 2):

- Procedure Builder для разработки моделей процессов
- Agent Toolkit для поддержки разработки агентов
- Task Manager для интеграции, визуализации и поддержки реализации
- Enterprise Ontology для коммуникации

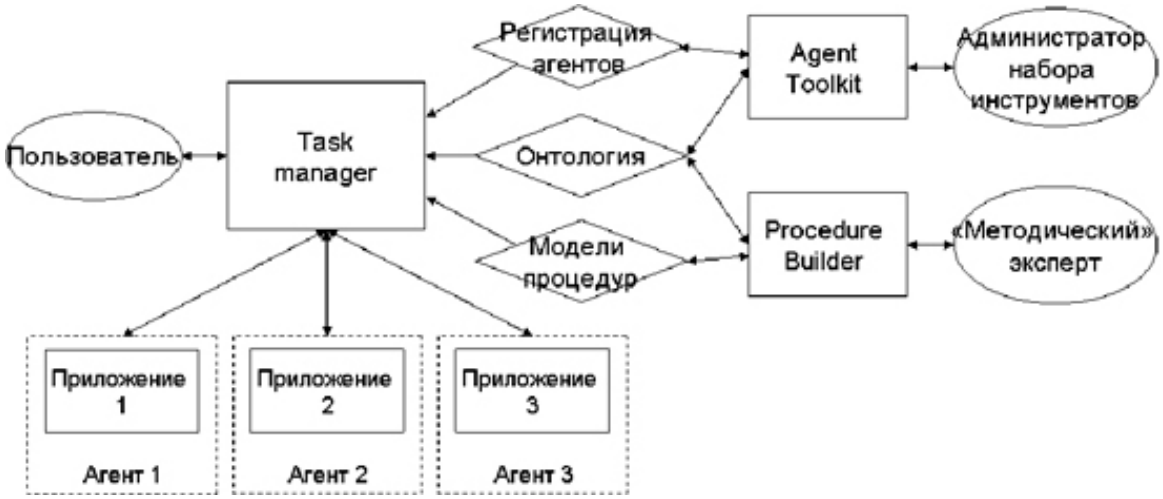

Рис. 2. Архитектура Enterprise Tool Set

Роль Онтологии Предприятий в Enterprise Project:

- Интеграция информации, используемой разными приложениями
- Описание функциональых возможностей пользовательских приложений
- Онтология в качестве семантики языка моделирования бизнес-процессов (по заявлениям самих участников, фактически, язык моделирования бизнес-процессов опирается в большей степени на свои собственные, не связанные с онтологией, понятия)
- Улучшение коммуникаций между сотрудниками организации

Онтология Предприятий состоит, главным образом, из набора Определенных терминов, явно представленных в онтологии, для которых даются определения и устанавливается взаимосвязь с другими терминами онтологии.

Все термины в онтологии попадают в 5 верхне-уровневых разделов, отражающих разные аспекты предприятия:

- Мета Онтология и Время
- Активность, План, Способность и Ресурс
- Организация
- Стратегия
- Маркетинг

В приводимой ниже таблице (табл. 1) перечислены все определенные в Онтологии Предприятий понятия, объединенные в основные группы.

Табл. 1. Общая модель Онтологии Предприятий

Активности и процессы	**Организация**	**Стратегия**	**Маркетинг**	**Время**
Активность	Лицо	Цель-назначение	Продажа	Ось времени
Спецификация активности	Машина	Иметь цель-назначение	Потенциальная продажа	Временной интервал
Выполнять	Корпорация	Планируемая цель-назначение	Для продажи	Момент времени
Выполненная спецификация активности	Партнерство	Держатель цели-назначения	Коммерческое предложение	
Т-Начала	Партнер	Стратегическая цель-назначение	Вендор	
Т-Окончания	Юридическое лицо	Задача	Фактический покупатель	
Предварительное условие	Подразделение	Виденье	Потенциальный покупатель	
Эффект	Управлять	Миссия	Покупатель	
Делатель	Делегировать	Цель-результат	Дистрибьютор	
Под-активность	Управленческая связь	Обеспечивать достижение	Продукт	
Полномочия	Право собственности	Стратегия	Запрашиваемая цена	
Владелец активности	Non-Legal собственность	Стратегическое планирование	Цена продажи	
Событие	Собственность	Стратегическое действие	Рынок	
План	Владелец	Решение	Переменная сегментации	
Под-план (sub-plan)	Актив	Допущение	Рыночный сегмент	
Планирование	Заинтересованная сторона	Критическое допущение	Маркетинговое исследование	
Спецификация процесса	Трудовой договор	Некритическое допущение	Бренд	
Способность	Доля / Акция	Фактор влияния	Имидж	
Умение	Собственник / Акционер	Критический фактор влияния	Характеристика	
Ресурс		Некритический фактор влияния	Потребность	
Распределение ресурсов		Критический фактор успеха	Потребность рынка	
Ресурс заменитель		Риск	Продвижение	
			Конкурент	

Далее для примера рассмотрены термины из двух разделов:
Активности и процессы и Мета-онтологии.

Активности и процессы

Центральным термином является АКТИВНОСТЬ (ACTIVITY). Он охватывает понятие всего, что предполагает реальное делание, в частности, включая действие. АКТИВНОСТЬ может иметь место в прошлом и происходить в настоящем. Этим термином можно также обозначать гипотетическую будущую АКТИВНОСТЬ.

Тем не менее, существует необходимость непосредственно указывать спецификацию или план АКТИВНОСТЕЙ. Это называется СПЕЦИФИКАЦИЯ АКТИВНОСТИ (ACTIVITY SPECIFICATION). Как и рецепт, она на некотором уровне детализации описывает одну или более АКТИВНОСТЕЙ. СПЕЦИФИКАЦИЯ ВЫПОЛНЕННОЙ АКТИВНОСТИ должна иметь нечто сделанное, соответствующее АКТИВНОСТИ.

Концепция АКТИВНОСТИ тесно связана с понятием ДЕЛАТЕЛЬ (DOER), который ВЫПОЛНЯЕТ (EXECUTES) СПЕЦИФИКАЦИЮ АКТИВНОСТИ, совершая заданные АКТИВНОСТИ. ДЕЛАТЕЛЬ может быть ЛИЦОМ (PERSON), ПОДРАЗДЕЛЕНИЕМ (ORGANISATIONAL UNIT) или МАШИНОЙ (MACHINE). Эти термины определены в секции Организация и вместе могут собирательно обозначаться как [ПОТЕНЦИАЛЬНЫЕ] ИСПОЛНИТЕЛИ ([POTENTIAL] ACTORS) (или АКТЕРЫ).

Возможность ПОТЕНЦИАЛЬНОГО ИСПОЛНИТЕЛЯ быть ДЕЛАТЕЛЕМ обозначается как СПОСОБНОСТЬ (CAPABILITY) или УМЕНИЕ (SKILL), если ДЕЛАТЕЛЬ – ЛИЦО. ИСПОЛНИТЕЛЬ может выполнять иные Роли в отношении АКТИВНОСТИ, такие, например, как ВЛАДЕЛЕЦ АКТИВНОСТИ (ACTIVITY OWNER).

С АКТИВНОСТЬЮ также тесно связан РЕСУРС (RESOURCE), представляющий собой нечто, могущее быть использованным или израсходованным АКТИВНОСТЬЮ. АКТИВНОСТЬ также может иметь выходы или ЭФФЕКТЫ (EFFECTS). АКТИВНОСТЬ связана с ВРЕМЕННЫМ ИНТЕРВАЛОМ (TIME INTERVAL). АКТИВНОСТЬ может занимать малое или большое время и быть простой или сложной. Сложная АКТИВНОСТЬ может раскладываться на множество ПОД-АКТИВНОСТЕЙ (SUB-ACTIVITIES).

СПЕЦИФИКАЦИЯ АКТИВНОСТИ совместно с ПЛАНИРУЕМОЙ ЦЕЛЬЮ-НАЗНАЧЕНИЕМ (INTENDED PURPOSE), определяемой в разделе Стратегия, называется ПЛАНОМ (PLAN). Понятие о возможности многократно ВЫПОЛНЯТЬ один и тот же ПЛАН описывается термином СПЕЦИФИКАЦИЯ ПРОЦЕССА (PROCESS SPECIFICATION).

Управление осуществлением АКТИВНОСТЕЙ важно для предприятия. Для этой цели мы определяем понятие ПОЛНОМОЧИЯ (AUTHORITY) ИСПОЛНИТЕЛЯ быть вправе выполнять одну или несколько АКТИВНОСТЕЙ (например, как указано в ПЛАНЕ).

Мета-Онтология и Время

Базовым понятием Мета-Онтологии является СУЩНОСТЬ (ENTITY). Это то, что охватывает все другие понятия. При построении Онтологии некоторые понятия могут рассматриваться как самостоятельные, независимые от других (например, ЛИЦО (PERSON)). Они непосредственно классифицируются как СУЩНОСТИ. Другие понятия более естественно видятся как ОТНОШЕНИЕ между двумя или более СУЩНОСТЯМИ (например, ПРОДАЖА). Поэтому, хотя ПРОДАЖА может юридически рассматриваться как СУЩНОСТЬ, ее точнее охарактеризовать как ОТНОШЕНИЕ.

В рамках ОТНОШЕНИЯ СУЩНОСТЬ может иметь РОЛЬ (например, Лицо может быть Клиентом в Продаже). Напротив, СУЩНОСТЬ может рассматриваться как АТРИБУТ или другая СУЩНОСТЬ (например, Дата рождения Лица).

Некоторые РОЛИ в ОТНОШЕНИЯХ специфичны в том, что исполнение этих РОЛЕЙ влечет за собой некие представления о действиях или суждениях (например, выполнение Активности или принятие Предположения). Мы определяем СУЩНОСТЬ, исполняющую такую РОЛЬ как АКТОР (ACTOR) (что является приблизительным синонимом к ‘агенту’ в других работах по онтологии). РОЛЬ, исполняемая АКТОРОМ, — это РОЛЬ АКТОРА. Только определенные СУЩНОСТИ могут играть такие РОЛИ, они являются ПОТЕНЦИАЛЬНЫМИ АКТОРАМИ. В настоящее время они включают в себя Лиц, ОРГАНИЗАЦИОННЫЕ ЕДИНИЦЫ и, в некоторых случаях, Машины.

Для удовлетворения потребностей множества пользователей и различных точек зрения в настоящее время и в будущем в Онтологии могут появляться или использоваться совместно с ней новые РОЛИ АКТОРОВ и вводиться новые ОТНОШЕНИЯ. Могут появляться и новые типы СУЩНОСТЕЙ АКТОРОВ, хотя возможно, менее часто.

Собирательно ситуация, характеризуемая одной или более СУЩНОСТЯМИ, участвующими в одном или более ОТНОШЕНИЯХ с одной или более другими СУЩНОСТЯМИ, определяется как СОСТОЯНИЕ ДЕЛ. СОСТОЯНИЕ ДЕЛ может иметь место, а может и нет (т.е. быть истинным или ложным).

Как упоминалось ранее, термины Онтологии не были точно определены в терминах этой Мета-Онтологии, если только это не казалось наиболее естественным выбором для конкретного термина. Тем не менее, Мета-Онтология подразумевалась в большей части работ по выбору терминов и определений. Отношения между терминами и Мета-Онтологией, как и ожидалось, стали более явными при последующем кодировании Онтологии в Ontolingua.

Время. Понятие времени не специфично для Предприятий, но они его используют. Мы не делали попыток переосмыслить существующие работы по представлению времени, вместо этого мы просто использовали их. Мы отмечаем, что при выполнении АКТИВНОСТИ следует говорить о ВРЕМЕННОМ ИНТЕРВАЛЕ. ВРЕМЕННОЙ ИНТЕРВАЛ определяется в терминах МОМЕНТОВ ВРЕМЕНИ, которые, в свою очередь, составляют ОСЬ ВРЕМЕНИ (TIME LINE).

The TOVE Project (Toronto Virtual Enterprise)

http://www.eil.utoronto.ca/enterprise-modelling/index.html
http://www.eil.utoronto.ca/enterprise-modelling/tove/

Целями проекта TOVE являлись:
1. Создание разделяемой терминологии (онтологии) для предприятия, которую каждый агент будет однозначно понимать и использовать.
2. Определить смысл каждого термина.
3. Ввести семантику в виде набора аксиом, которые позволяют TOVE автоматически выводить ответы на вопросы «здравого смысла» («common sence») о предприятии.
4. Определить символьные обозначения для терминов и понятий.

Модель TOVE является многоуровневой, охватывающей концептуальный уровень, обобщенный и уровень приложений. Обобщенный уровень и уровень приложений в свою очередь также являются стратифицированными и состоящими из микротеорий, охватывающих, например, активность, время, ресурсы, ограничения и т.д. на обобщенном уровне. Экспериментальная система TOVE обеспечивает среду для анализа онтологий предприятия, обеспечивает модель предприятия и инструменты для навигации, визуализации, симуляции и дедуктивных запросов. Критическим требованием в проекте TOVE является создание экземпляра модели для конкретного предприятия.

Модели TOVE автоматически создаются как побочный продукт функции проектирования предприятия. Именно в результате определения целей, действий предприятия формируется модель. Однако для того чтобы успешно сформировать модель, методы моделирования должны быть более четкими, с точки зрения спецификации целей, активностей, ограничений, ресурсов и т.д., чем в существующих сейчас инструментах моделирования.

Язык спецификации процессов — Process Specification Language (PSL)

Язык спецификации процессов является развитием результатов, полученных в проекте TOVE. В инициативах по реинжинирингу бизнес-процессов или по интеграции предприятий критическим вопросом является возможность обмениваться и связывать разнообразные модели процессов. Целью Process Specification Language (PSL) Проекта является разработка формата обмена для передачи описаний процессов между различными системами бизнес-моделирования и управления бизнес-процессами, такими как системы work-flow, системы симуляции, инструменты бизнес-реинжиниринга и репозитарии процессов. PSL выступает как единый формат, обеспечивающий интероперабельность, заменяющий постоянную разработку ad hoc трансляторов для каждой пары описаний процессов. PSL дает приложениям общее понимание понятий, которыми необходимо обмениваться.

Целевой аудиторией PSL являются производители, у которых возрастает потребность в обмене процессной информацией между различными приложениями.

В отличие от языков моделирования процессов, PSL является языком обмена процессной информацией между приложениями. Например, он позволит передать модель процесса из приложения, использующего IDEF3, в приложение, использующее сети Petri.

PSL основан на формальной онтологии. Все понятия PSL формально определены на основе Knowledge Interchange Format (KIF) для устранения двусмысленности, которая часто встречается в приложениях, описывающих процессы. Онтология состоит из ядра и серии модулей, основанных на Ядре. PSL Ядро — модуль, который включает выскоуровневые базовые понятия, присущие описанию процесса. Каждый модуль проясняет Ядро и включает набор понятий, связанных с определенной областью описания процесса (ресурсы, роли, время). Модули строятся друг на друге, что делает PSL онтологию похожей на сеть модулей, в основании каждого из которых лежит Ядро. Подробнее см. http://www.mel.nist.gov/psl/index.html.

Проект SUPER

http://ip-super.org/

В качестве наиболее передового и крупного проекта в области развития технологий моделирования организаций и управления бизнес-процессами можно выделить проект SUPER (Semantics Utilized for Process Management within and between Enterprises / Использование семантики для управления процессами внутри и между предприятиями). Основной целью проекта SUPER является перевод управления бизнес-процессами (BPM) с уровня информационных технологий на бизнес-уровень. Такая цель требует доступности системы управления процессами на уровне понятий бизнес экспертов, способом представления которых выступают онтологии. Авторы называют свой подход семантическим (semantic) управлением бизнес-процессами (SBPM) [Hepp, 2005].

В рамках проекта планируется разработка инструмента, поддерживающего анализ, изменение и создание бизнес-процессов, направленного на повышение уровня гибкости и адаптивности организаций. Планируемый инструмент будет основан на семантической аннотации артефактов, относящихся к управлению бизнес-процессами (операции

процессов, сервисы и т.п.). Такая аннотация позволит создавать более эффективные запросы и осуществлять автоматизированный вывод, что, в свою очередь, позволит пользователям осуществлять «семантический» поиск компонент бизнес-процессов, «семантическое» составление бизнес-процессов и «семантическое» взаимодействие бизнес-процессов.

«Семантический» поиск элементов бизнес-процессов поддерживает бизнес-экспертов на этапе моделирования процессов, упрощая повторное использование существующих компонент. Для решения данной задачи создается онтологическая система, которая позволит описывать бизнес-процессы и их компоненты, а бизнес-экспертам формировать необходимые запросы к репозитарию для поиска существующих компонент бизнес-процессов. «Семантическое» составление бизнес-процессов направлено на предоставление бизнес-экспертам возможности автоматической генерации исполнимых бизнес-процессов по их концептуальным моделям (Рис. 3):

Рис. 3. Композиция (составление) процесса на основе семантически аннотированных сервисов и процессов

«Семантическое» взаимодействие бизнес-процессов поддерживает «Семантическое» составление бизнес-процессов, путем интеграции процессов, созданных различными участниками для создания единого процесса, требующего сотрудничества нескольких сторон. Использование предлагаемых технологий позволит:

- Повысить качество создаваемых процессных моделей, благодаря повторному использованию существующих оптимизированных компонент бизнес-процессов.
- Сократить время моделирования бизнес-процессов, устраняя «изобретение колеса».

SBPM-инструмент ориентирован на совместимость с существующими инструментами и стандартами организационного моделирования. В частности, процессы, представленные на BPEL, BPMN или EPC, должны «читаться» («импортироваться») SBPM-инструментом. Также процессы, созданные в SBPM-инструменте, должны экспортироваться в BPEL для исполнения в соответствующих инструментах (BPEL Engines) — см. рис. 4:

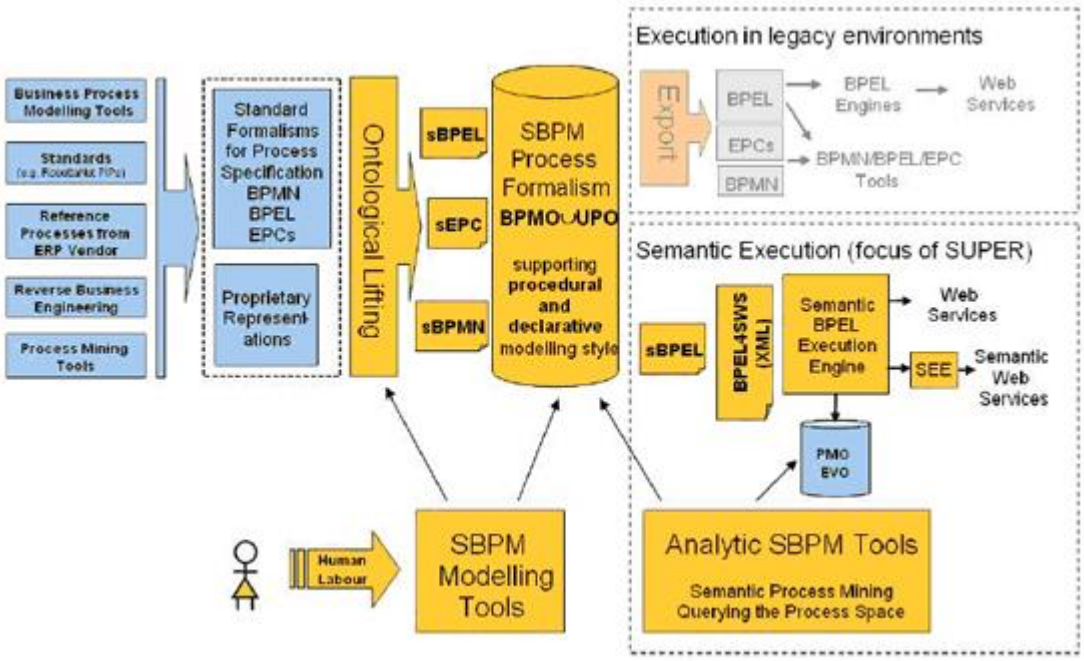

Рис. 4. Схема функционирование SBPM

Основная идея в SBPM-подходе — это сочетание системы семантических веб-сервисов, онтологической инфраструктуры и методологии и инструментов управления бизнес-процессами для создания единой технологии, которая обеспечит новый уровень перевода бизнес-требований в область фактического исполнения процессов.

В настоящий момент проектируемая система онтологий («SUPER Set of Ontologies for Business Process Management») состоит из 8-ми разделов и выглядит следующим образом [Hepp, 2007]:

1. Процессы
1.1. Верхнеуровневая процессная онтология (Upper Process Ontology)
Процесс (Process)
Активность (Activity)
Предположение (Assumption)
Предусловия (Pre-condition)
Пост-условие (Postcondition)
Эффект (Effect)
1.2. Онтология ЯОМ BPEL (sBPEL ontology, an ontology version of BPEL)
1.3. Онтология ЯОМ BPMN (sBPMN, an ontology version of BPMN)
1.4. Онтология ЯОМ EPC (sEPC, an ontology version of EPCs)
2. Организация и ресурсы
2.1. Верхнеуровневая онтология организации (Upper Organizational Ontology)
Организация (Organization)
Роль (Role)
Задача (Task)
Дивизион (Division)
Ресурс (Resource)
Сотрудник (Employee)
Машина или Система (MachineOrSystem)
Нематериальный ресурс (IntangibleResource)
Навыки и Способности (Skill and Capability)
2.2. Онтология Организации для бизнеса (Business Organization Ontology)

2.3. Онтология ресурсов для бизнеса (Business Resources Ontology)
3. Онтология бизнес функций
3.1. Маркетинг (Marketing)
3.2. Человеческие ресурсы (Human Resources)
3.3. Операции (Operations)
...
4. Данные
4.1. Верхнеуровневая модель данных организации (Enterprise Data Upper Ontology)
Система управления базой данных (DBMS)
База данных (DataBase)
Схема базы данных (DataBaseTable)
4.2. Онтология транзакцмй данных (Transactional Data Ontology)
4.3. Онтология кастомизации данных (Customizing Data Ontology)
5. Онтология обеспечения и потребления
6. Онтология правил и ограничений предприятия
7. Онтология стратегии
Цель (Goal)
Подцель (SubGoal)
8. Предметные онтологии

Понятийное и объектное моделирование властных структур на региональном уровне ИПУСС РАН

В Институте проблем управления сложными системами РАН и научно-производственных компаниях-партнерах Института накоплен значительный опыт по использованию онтологического подхода в задачах информационной поддержки административной деятельности на региональном уровне (см., например, [Батищев и др., 2003], [Виттих и др., 2004], [Волхонцев и др., 2005], [Хасаев и др., 2003], [Виттих и др., 2006]).

Этот опыт оказался востребованным и в связи проведением в настоящее время административной реформы на федеральном и региональном уровнях [Концепция, 2005].

Одним из направлений повышения эффективности деятельности органов исполнительной власти (а это, разумеется, важнейшая цель административной реформы) видится представление организационных структур с фиксацией всех связей в строгом, непротиворечивом виде, обеспечивающем возможность их наглядного, лаконичного, но одновременно полного обозрения. Такого эффекта невозможно добиться при помощи обычного текстового представления информации. Поэтому, на наш взгляд, лишь внедрение некоторого релевантного формализованного описания организационных структур даст основания для четкого отслеживания и совершенствования функций, структуры и взаимоотношений органов исполнительной власти, а в рамках получающего все большее распространение в административных структурах проектного управления — систему управления конкретными проектами.

Кроме того, одним из региональных приоритетов в области реформирования государственного управления является повышение роли взаимодействия в процессах принятия решений: в организации процессов управления в Самарской области предполагается взаимодействие органов власти, предпринимателей, общественных деятелей и жителей области, которые готовы реализовывать свои гражданские права и обязанности [Титов и др., 2006]. Необходимым условием такого разнородного взаимодействия является наличие общего понятийного базиса, на основе которого можно описывать предметные и проблемные области, специфицировать организационные структуры по решению поставленных задач, осуществлять принятие управленческих решений.

Решение отмеченных проблем связывается с построением онтологий, характеризующих различные аспекты системы исполнительной власти (на примере системы исполнительной власти Самарской области), и на этой основе — объектных моделей соответствующих предметных областей. Для представления и обозрения концептуальных и объектных структур предлагается использовать специально разработанный программный инструментарий.

Понятийные и объектные модели системы исполнительной власти

Конструирование всех онтологий и объектных моделей выполнялось с помощью разработанной в ИПУСС РАН общецелевой системы объектно-ориентированного моделирования gB [Смирнов, 1999], [Смирнов, 2004].

Первой была разработана онтология структуры системы исполнительной власти Самарской области. При построении этой понятийной модели были использованы Устав Самарской области, положения и регламенты деятельности органов исполнительной власти Самарской области, перечень органов исполнительной власти Самарской области, не являющихся министерствами Самарской области (иных органов исполнительной власти Самарской области), другие нормативно-правовые акты, мнения экспертов. В онтологии представлено более 45 классов объектов, из которых – 27 листовых, 10 видов отношений между объектами. В частности, модель включает понятия «Губернатор», «Вице-губернатор», «Правительство», «Аппарат Правительства», «Министерство», «Иной орган исполнительной власти» и т.д. В рассматриваемой редакции модели детализация производится до уровня органов исполнительной власти, не затрагивая их внутреннюю структуру. Исключение сделано лишь для организационной структуры аппарата Правительства.

При решении множества конкретных задач востребовано описание еще двух аспектов системы исполнительной власти, касающихся должностной иерархии чиновников и персональных данных граждан, занимающих те или иные должности. Поэтому была разработана онтология должностей системы исполнительной власти Самарской области (более 90 классов объектов и 6 видов отношений), а также элементарный вариант онтологии персоналий.

Созданная понятийная база позволила осуществить описание конкретных объектов (точнее систем объектов) в актуализированных предметных областях, т.е. построить объектные модели исследуемой системы, где представлены экземпляры понятий и отношений. Например, в объектной модели структуры исполнительной власти Самарской области, которая включает сотни объектов с множественными связями десяти видов, понятию «Министр-руководитель» (одна из штатных категорий, конкретизирующих обобщенное понятие «Состоящие в ранге министра») отвечают «министр гуманитарного и социального развития», «министр здравоохранения», «министр образования» и т.п.

Обозреватель моделей

Потенциально многообещающая прагматика использования построенных понятийных и объектных моделей системы исполнительной власти в простейшем варианте заключается в возможности удобного обозрения соответствующих структур с целью изучения, понимания и получения справочной информации. Очевидно, что для более сложных приложений созданных моделей, компонента визуализации также будет играть ключевую роль.

Фундаментальный анализ проблем визуализации графов (а именно эта абстракция чаще других применяется для представления информации, которую можно промоделировать в виде объектов и связей между ними), включая обзор компьютерных инструментальных систем визуализации, дан в [Касьянов и др., 2003]. Выполненное нами исследование этой проблематики позволило сделать вывод о необходимости разработки

методов и инструментов, адекватных специфике онтологического подхода при моделировании сложных систем.

Эта специфика заключается не только и не столько в необходимости представления особенных компонентов рассматриваемых структур (т.е. объектов, их атрибутов, связей и классов) или в способе представления больших объектных структур. Глубинная специфика онтологического подхода состоит, на наш взгляд, в необходимости совместного, одновременного манипулирования в приложениях несколькими такими структурами, которые совместно описывают многоаспектную семантику моделируемой предметной области [Смирнов, 1999], [Смирнов, 2004]. Это понимание вполне отвечает представлениям о многомодельных системах в общей парадигме гибридных методов решения задач [Емельянов и др., 2003].

Обозреватель объектных моделей, обладающий требуемой уникальной функциональностью, разработан как специальное приложение инструментальной системы gB. Для структуризации обозрения были объединены идеи методов блочного представления объектных моделей [Виттих и др., 2004] и локально-радиального обзора свойств избранного объекта, который предложен для семантических сетей М. Штефанером из Института прикладных информационных технологий Общества им. Фраунгофера (см. http://www.der-mo.net/). Приоритетная новизна разработанного обозревателя состоит в наличии средств «гиперперехода» между сосуществующими в обозреваемом приложении контекстами моделирования. С целью увязывания различных контекстов моделирования сложной системы предложено динамически внедрять в обозреваемые диаграммы объектов и связей семантически релевантные содержанию диаграммы псевдообъекты — «класс обозреваемого объекта» и «лицо многоликого объекта задачи» (о необходимости введения в приложения «многоликих» объектов задачи см. [Смирнов, 2004]). Применение к псевдообъекту стандартных для обозревателя способов навигации в объектном пространстве влечет «гиперпереход» к обозрению объектной модели сосуществующего контекста моделирования.

Прагматика полученного в работе результата заключается в том, что в сфере административного управления регионом, традиционно опирающемся на текстовые описания сложных концептуализаций, предложено использовать понятийные модели (онтологии), допускающие возможность формализованного анализа структур и каналов взаимодействия при принятии решений. Построенные модели могут служить основой для упорядочивания взаимодействия не только должностных лиц и органов исполнительной власти, но гражданского общества в целом.

Конфигурирование услуг электронного государства на основе онтологий (Ontology-enabled e-Gov Service Configuration)

OntoGov — проект Евро-союза, направленный на разработку семантически обогащенной платформы на основе онтологий, которая обеспечит последовательные (непротиворечивые):

- построение,
- переконфигурацию,
- эволюцию услуг электронного государства.

Основными задачами проекта являются:

- Обеспечение чиновников средствами, позволяющими им увидеть существующую модель оказания услуг и легко изменить ее в случае необходимости.
- Обеспечение всех участников, вовлеченных в жизненный цикл оказания государственных услуг, улучшенными знаниями по построению, переконфигурации и эволюции услуг электронного правительства.

Проект должен обеспечить более эффективное прохождение следующей цепочки: Изменение закона → Изменение деятельности исполнительной власти → Изменения в

программных приложениях → Изменения в услугах, оказываемых конечному потребителю. Архитектура среды, реализующей данную идею, представлена на рис. 5:

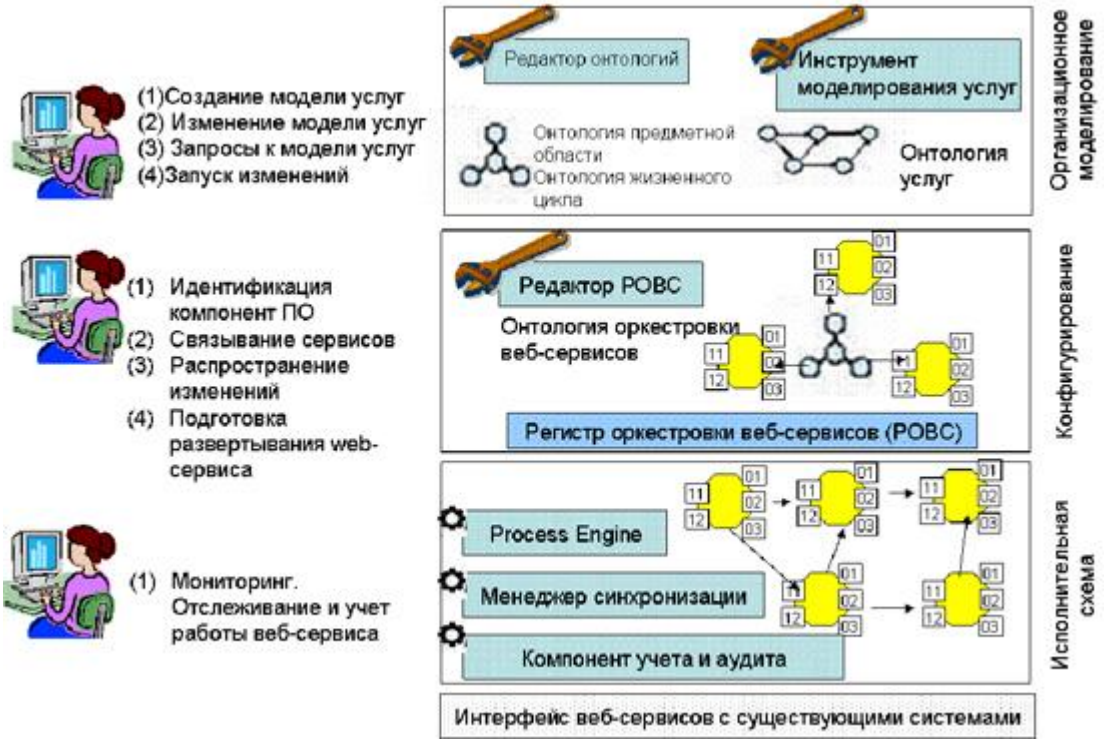

Рис. 5. Архитектура среды конфигурирования сервисов

Для поддержки логической архитектуры на основе анализа Semantic Web Services (i.e. OWL-S and WSMO) был определен набор онтологий, которые можно обобщить в 3 следущие группы:

- Мета-онтологии
- Онтологии предметных областей
- Административные онтологии

Мета-онтологии определяют схему, то есть язык для описания услуг электронного правительства. Онтологии предметных областей моделируют конкретные услуги электронного правительства и описывают данные, необходимые для этих услуг. Основная онтология в данной группе — это так называемая Онтология Услуг, которая отражает услуги электронного правительства. Поскольку целью проекта является улучшение управления услугами электронного государства, были созданы административные онтологии. Подробнее см. http://www.ontogov.com/.

Заключение

Резюмируя описание примеров использования онтологий в организационном моделировании, можно сказать, что результаты проектов Enterprise Project, TOVE, PSL, SUPER не удовлетворяют необходимым требованиям, предъявленным к системам организационного моделирования. В первую очередь, это связано с тем, что предлагаемые в них модели и методы не дают комплексного решения. Они либо не интегрированы с существующими инструментами автоматизированной поддержки организационного проектирования (TOVE, PSL), либо поддерживают решения частных задач (автоматизация бизнес-процессов организации — Enterprise Project, поддержка рассуждений на основе формальной организационной модели — TOVE), либо покрывают только часть области организационного проектирования (например, моделирование только бизнес-процессов — PSL). Проект SUPER находится в начальной стадии, а также он, в первую очередь, ориентирован на задачу автоматизации бизнес-процессов организаций.

Литература

1. Abecker, A.; Apostolou, D.; Hinkelmann, K.; Probst, F.; Stojanovic, L.; Tambouris, T. Ontology-enabled E-Government Service Configuration - The OntoGov Approach. In: Wimmer, Maria A. (Ed.): e-Gov Days: state-of-the-art 2004. Tagungsband zu den dritten e-Gov Days des Forums eGovernment. Wien: OCG 2004.
2. Bock, C., Gruninger, M., "PSL: A Semantic Domain for Flow Models," Software and Systems Modeling Journal, 2005.
3. Fox, M.S. "The TOVE Project: A Common-sense Model of the Enterprise", Industrial and Engineering Applications of Artificial Intelligence and Expert Systems, Belli, F. and Radermacher, F.J. (Eds.), Lecture Notes in Artificial Intelligence # 604, 1992. Berlin: Springer-Verlag, pp. 25-34.
4. Fox M., Barbuceanu M., Gruninger M.; Lin J. An Organization Ontology for Enterprise Modelling. Simulating Organizations: Computational Models of Institutions and Groups, Menlo Park CA: AAAI/MIT Press, pp. 131-152. 1997.
5. Gomez-Perez A. Ontologies: Theory, methods and tools. Tutorial. The Fourth Summer School on Ontological Engineering and the Semantic Web, 2006 (SSSW'06).
6. Gruninger M., Atefi K., Fox, M., Ontologies to support process integration in enterprise engineering, Computational and Mathematical Organization Theory, 6, pp. 381-394, 2000.
7. Hepp, Martin et al.: Semantic Business Process Management: A Vision Towards Using Semantic Web Services for Business Process Management. IEEE International Conference on e-Business Engineering (ICEBE 2005). Beijing, China, October 18-20, 2005, pp. 535-540.
8. Hepp M., Roman D. An Ontology Framework for Semantic Business Process Management, Proceedings of Wirtschaftsinformatik 2007, February 28 - March 2, 2007, Karlsruhe
9. Stader J., Results of the Enterprise Project // Proceedings of Expert Systems '96, the 16th Annual Conference of the British Computer Society Specialist Group on Expert Systems, Cambridge, UK, December 1996.
10. Uschold M., King M., Moralee S. and Zorgios Y. The Enterprise Ontology AIAI, The University of Edinburgh, 1997.
11. Батищев С.В. , Виттих В.А., Генералова Л.Д. и др. Онтология социокультурных ресурсов Самарской области // Проблемы управления и моделирования в сложных системах: Труды V международной конф. (Самара, 17-21 июня 2003 г.). — Самара: СамНЦ РАН, 2003.- С. 402-409.
12. Виттих В.А., Ситников П.В., Смирнов С.В. Понятийное и объектное моделирование властных структур // Труды 10-й Национальной конференции по искусственному интеллекту с международным участием (КИИ-06), 26-28 сентября 2006.
13. Виттих В.А., Иванова Л.А. , Королева Е.Н. и др. Проблемы онтологической спецификации объектов региональной экономики // Проблемы управления и моделирования в сложных системах: Труды VI международной конф. (Самара, 14-17 июня 2004 г.). — Самара: СамНЦ РАН, 2004. — С. 322-327.
14. Волхонцев Д.В. , Гриценко Е.А. , Зубайдулаева Е.Ю. и др. Разработка конструктора нормативно-правовой базы знаний для социальной сферы // Проблемы управления и моделирования в сложных системах: Труды VII международной конф. (Самара, 27 июня-2 июля 2005 г.). — Самара: СамНЦ РАН, 2005. С. 357-365.
15. Гаврилова Т. Онтологический подход к управлению знаниями при разработке корпоративных информационных систем. // Ж. "Новости искусственного интеллекта", N2, 2003. - с.24-30.
16. Емельянов В.В., Курейчик В.В., Курейчик В.М. Теория и практика эволюционного моделирования. — М.: ФИЗМАТЛИТ, 2003.
17. Касьянов В.Н., Евстигнеев В.А. Графы в программировании: обработка, визуализация и применение. — СПб.: БХВ-Петербург, 2003.
18. [Концепция, 2005] Концепция административной реформы в Российской Федерации в 2006–2008 годах и план мероприятий по проведению административной реформы в

Российской Федерации в 2006–2008 годах (одобрены распоряжением Правительства РФ от 25 октября 2005 г. № 1789-р).
19. Смирнов С.В. Открытая архитектура инструментальных средств моделирования сложных систем // Проблемы управления и моделирования в сложных системах: Труды международной конф. (Самара, 15-17 июня 1999 г.). — Самара: СамНЦ РАН, 1999. - С. 59-66.
20. Смирнов С.В. Онтологии в прикладных интеллектуальных системах: прагматический подход // Девятая Национальная конф. по искусственному интеллекту с международным участием КИИ-2004 (Тверь, 28 сентября-2 октября 2004 г.): Труды конф. Т. 3. — М.: ФИЗМАТЛИТ, 2004.. - С. 1059-1067.
21. Титов К.А., Виттих В.А. Концепция организации процессов управления в регионе // Проблемы управления и моделирования в сложных системах: Труды VIII международной конф. (Самара, 24-28 июня 2006 г.). — Самара: СамНЦ РАН, 2006. С. 305-313.
22. Хасаев Г.Р. , Виттих В.А., Иванова Л.А. и др. Региональная экономика как объект онтологического анализа // Известия Самарского научного центра РАН. 2003. Т. 5. № 1. — С. 74-82.
23. [Эталонные модели, 2006] «Эталонные модели организации деятельности в государственном секторе» // Отчет по научно-исследовательской работе выполненной сотрудниками АНО КМЦ «Бизнес-Инжиниринг» совместно с ИПГМУ ВШЭ, 2006 г.

АвиаОнтология: анализ современного состояния ресурса

Н.В. Лукашевич НИВЦ МГУ; АНО Центр информационных исследований, louk@mail.cir.ru
Невзорова НИИММ им. Н.Г. Чеботарева; Казанский государственный педагогический университет, Olga.Nevzorova@ksu.ru
http://www.dialog-21.ru/Archive/2004/Lukashevich.htm Доклады международной конференции Диалог 2004

Статья посвящена анализу современного состояния нового информационного ресурса АвиаОнтологии. Онтология была разработана на основе технологии создания больших и сверхбольших онтологий и тезаурусов для различных областей в НИВЦ МГУ. В настоящее время АвиаОнтология включает около 1600 понятий и 4700 терминов. В статье мы рассмотрим структурные характеристики онтологии и визуальную модель ее представления, выполненной с помощью системы “OntoEditor”. Данное исследование выполнено при поддержке Российского Фонда Фундаментальных исследований, грант № 02-07-90279.

1. Введение

Информационный ресурс АвиаОнтология является лингвистической онтологией, разработанной для специальной прикладной области. Данная прикладная область включает знания об авиации и специализирована на знаниях об информационных событиях и процессах в авиационной практике. Прежде всего это касается событий и процессов функционирования бортовой аппаратуры, а также экипажа в различных полетных режимах.
Необходимость разработки онтологии была вызвана прикладными задачами автоматической обработки специализированных текстов данной предметной области. Одним из приложений АвиаОнтологии является система автоматического анализа специальных технических текстов типа «Логика работы…» [1]. Задачи приложения определили специализацию АвиаОнтологии, которая связана, в первую очередь, с расширенным описанием области процессов обмена и передачи информации при решении задач авиационной тематики.

К настоящему времени АвиаОнтология содержит 1600 понятий, 4700 текстовых единиц. Проектирование подобной модели является сложной экспертной задачей, требует

взвешенного подхода к выбору формализма описания. В качестве базовой технологии для проектирования прикладной онтологии была выбрана технология проектирования больших и сверхбольших онтологий и тезаурусов для различных предметных областей [2], разработанная в рамках проекта Университетская информационная система РОССИЯ (www.cir.ru), поддерживаемая Научно-исследовательским вычислительным центром МГУ им. М.В.Ломоносова и АНО Центр информационных исследований.

2. Структура АвиаОнтологии

Разработка АвиаОнтологии осуществляется на основе электронной коллекции текстов данной предметной области, которая в настоящий момент составляет свыше 100 Мб.

Основными текстовыми источниками знаний о рассматриваемой предметной области являются серия специальных книг, отобранных экспертами в данной области, и текстовая информация из Интернет по отдельным разделам предметных знаний.

АвиаОнтология представляет иерархическую сеть понятий, и ее проектирование осуществлялось в несколько этапов.

На первом этапе осуществлялось построение "терминологического портрета" предметной области на основе автоматических методик. Специализированные алгоритмы автоматического выделения терминологических словосочетаний, в том числе и многословных терминов, в текстах электронной коллекции описаны в [3]. Использование различных фильтров при обработке списка кандидатов в термины (около 14 000 слов и словосочетаний) позволило автоматически выделить около 700 основных терминов предметной области.

Терминология любой предметной области содержит как специфические термины, употребляемые только в данной области или в ряде близких областей, так и достаточно общеизвестные термины. В данной области таким общеизвестными терминами являются летчик, самолет, истребитель, оружие, боевые действия, атака и многие другие. Это позволяет не начинать разработку модели предметной области с нуля, а использовать знания, описанные в более общих лингвистических и терминологических ресурсах. В качестве такого ресурса был использован тезаурус РуТез, разработанный АНО Центр информационных исследований.

Тезаурус представляет собой иерархическую сеть понятий, каждое из которых имеет ряд текстовых вариантов (способов языкового выражения) и совокупность отношений с другими понятиями тезауруса. Объем ресурса на момент проектирования онтологии составлял 97 тысяч слов и словосочетаний, уложенных в 42 тысячи понятий. Между понятиями вручную установлено более 160 тысяч связей. По свойствам транзитивности и наследования выводится более 1200000 связей между понятиями.

Наличие большого общезначимого лингвистического ресурса позволило сопоставить данный ресурс с текстами в предметной области, выделить описанные в тезаурусе типы знаний (понятия, синонимы, связи между понятиями), перенести их в специальную рабочую область как основу для построения прикладной онтологии.

Списки набранных слов и словосочетаний предметной области были сопоставлены с терминами Общественно-политического тезауруса. Если сопоставление очередного словосочетания было успешно, соответствующее понятие Общественно-политического тезауруса вместе со всеми терминами, выражающими его в тексте, копировалось в модель предметной области. На следующем шаге были скопированы все отношения Общественно-политического тезауруса между скопированными понятиями.

Кроме того, было выполнено замыкание отношений - выбирались не только те понятия, которые упомянуты в текстах предметной области непосредственно, но и те понятия тезауруса, которые находятся на концептуальных путях между упомянутыми в текстах понятиями - если понятие B является вышестоящим для понятия A, понятие C является вышестоящим для понятия B, причем понятия A и C были скопированы в

предметную область, то и понятие В также копируется в модель предметной области вместе со своими терминами-вариантами и релевантными отношениями.

Безусловно, перенесенные понятия и отношения требуют тщательного дополнительного тестирования для задания настройки на конкретную область. Например, в результате переноса среди терминов может встретиться синонимический вариант, который маловероятен в данной области, например, словосочетание винтокрылая машина как синоним слова вертолет вряд ли может встретиться в профессиональной технической области. Поэтому был осуществлен ручной контроль перенесенных терминов.

В существующей версии Авиа-Онтологии около 500 понятий, 1000 синонимов и 1000 отношений было перенесено из тезауруса РуТез, что обеспечило быстрый рост Авиа-Онтологии и существенно сократило время разработки нового ресурса.

Тестирование АвиаОнтологии на специальных текстах, при которых анализировалось покрытие текстов терминами из онтологии, показало, что для задач прикладного использования АвиаОнтологии требуется ввести в онтологию важные общезначимые слова, такие как необходимость, вероятность, условие и т.п. Группа из 130 таких понятий была добавлена в АвиаОнтологию.

Следующий этап проектирования онтологии - проектирование сети отношений онтологических концептов. Ключевой проблемой данного этапа является выбор списка отношений между концептами. В качестве базовой системы отношений была использована система отношений тезауруса РуТез, которая включает иерархическое отношение (ВЫШЕ-НИЖЕ), отношение ЧАСТЬ-ЦЕЛОЕ, отношения онтологической зависимости (АСЦ, АСЦ1, АСЦ2).

Отношения ЧАСТЬ – ЦЕЛОЕ – используется для описания как традиционных частей, так и участников ситуаций, свойств.

Например,

Самолет заправщик

Целое ДОЗАПРАВКА В ВОЗДУХЕ

ФЮЗЕЛЯЖ

Целое САМОЛЕТ

ЛЕТНО-ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ

Целое ЛЕТАТЕЛЬНЫЙ АППАРАТ

Характерной онтологической особенностью всех этих достаточно традиционных видов отношений является то, что ЦЕЛОЕ (носитель свойства, ситуация) обычно онтологически зависит [4] от своих ЧАСТЕЙ (свойств, участников), имеет таких ЧАСТЕЙ (свойств, участников) более одного, и эти ЧАСТИ (свойства, участники) могут онтологически не зависеть друг от друга.

Для обеспечение транзитивности логического вывода по отношению ЧАСТЬ – ЦЕЛОЕ требуется, чтобы ЧАСТИ также онтологически зависели от ЦЕЛОГО. Чаще всего такое описание возможно. Например, не следует в АвиаОнтологии считать, что ДВИГАТЕЛЬ это ЧАСТЬ САМОЛЕТА, двигатели бывают также, например, в ракетах. Это в данном случае и означает, что существование понятия ДВИГАТЕЛЬ не зависит от существования понятия САМОЛЕТ. Таким образом, необходимо выделить вид двигателя АВИАЦИОННЫЙ ДВИГАТЕЛЬ, который и будет описан как ЧАСТЬ понятия САМОЛЕТ.

Для описания других отношений онтологической зависимости используется несимметричная ассоциация АСЦ1-АСЦ2 (АСЦ1="зависит_от", АСЦ2="главное_для");

Симметричная ассоциация используется для сходных по смыслу понятий.

Приведем пример словарной статьи Авиа-Онтологии, содержащей все типы отношений, с пометкой 'син' даны синонимы понятия:

ПУСК РАКЕТЫ

син запуск ракеты

син	запустить ракету
син	отстрел ракеты
син	применение ракет
син	применение ракетного вооружения
син	пуск ракеты
син	пустить ракету
син	ракетная атака
син	ракетный удар
ВЫШЕ	ПРИМЕНЕНИЕ ОРУЖИЯ
НИЖЕ	ЭФФЕКТИВНЫЙ ПУСК РАКЕТЫ
ЧАСТЬ	ДАЛЬНОСТЬ ПУСКА РАКЕТЫ
ЧАСТЬ	ЗОНА ВОЗМОЖНОГО ПУСКА
ЧАСТЬ	МАКСИМАЛЬНО ДОПУСТИМАЯ ПЕРЕГРУЗКА ПРИ ПУСКЕ
АСЦ1	РАКЕТА
АСЦ2	ВЫХОД В ЗОНУ ВОЗМОЖНОГО ПУСКА
АСЦ2	КОМАНДА «ПУСК РАЗРЕШЕН»
АСЦ2	ОШИБКА ПУСКА
АСЦ2	УПРАВЛЕНИЕ РАКЕТОЙ

3. Визуализация АвиаОнтологии

Исследование характеристик АвиаОнтологии было выполнено с помощью инструментальной системы визуального проектирования онтологий «OntoEditor» разрабатываемой в НИИ математики и механики им. Н.Г. Чеботарева г. Казань, руководитель проекта О.А. Невзорова. Проектирование информационных ресурсов типа онтологий на основе специализированной программной системы «OntoEditor» осуществляется на основе визуальной технологии, позволяющей эксперту или инженеру по знаниям вводить понятия, синонимические ряды понятий, связи между понятиями и отображать введенную информацию в графическом режиме.

Инструментальная система визуального проектирования «OntoEditor» является специализированной СУБД. Система предназначена для ручного редактирования онтологий, хранящихся в реляционной базе данных в определенном формате, а также обслуживания запросов пользователей и внешних программ.

Инструментальная система позволяет:

• добавлять, изменять и удалять отдельные записи БД;

• автоматически корректировать данные при удалении конкретных записей (например, удалять отношения стертого концепта);

• поддерживать ведение нескольких онтологий, в том числе смешанных (например, с общими списками типов отношений, классов, синонимов и др.);

• импортировать онтологии различных форматов данные из внешних баз данных (механизмы импорта разрабатываются на конкретную базу данных);

• вести обработку онтологий в табличной и графической формах;

• поддерживать иерархическую структуру классов;

• выделять по заданному фильтру определенные подмножества редактируемой онтологии;

• вести автоматическую статистику по объектам онтологии;

• осуществлять поиск цепочек отношений концептов с заданными свойствами;

• выводить требуемую информацию на печать или в текстовый файл в заданных формах отчета в кодировках ANSI и ASCII;

• обрабатывать внешние запросы с использованием механизма обмена DDE (Dynamic Data Exchange).

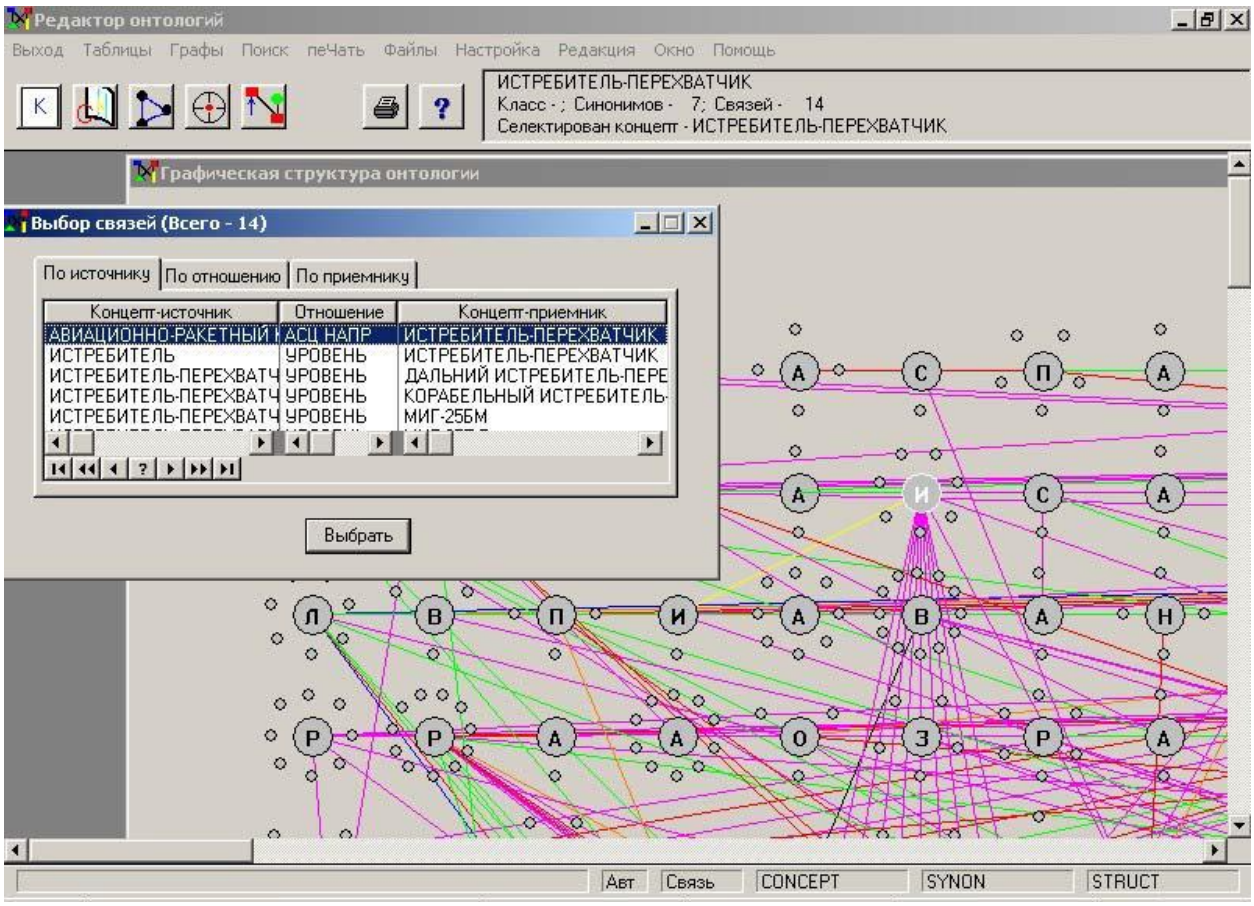

Рис.1. Визуальный образ АвиаОнтологии

Система поддерживает многооконный интерфейс и снабжена развитой системой подсказок, а также механизмами поиска конкретной записи.

На рис. 1 показан визуальный образ АвиаОнтологии с селектированным концептом ИСТРЕБИТЕЛЬ (выделен цветом на форме) и окном отображения связей селектированного концепта. Синонимы концепта размещаются на окружности с центром-концептом, типы отношений выделяются цветом.

С помощью инструментальной системы визуального проектирования «OntoEditor» были получены графические образы АвиаОнтологии с различными структурными характеристиками. Ранжирование АвиаОнтологии по различным критериям (по числу синонимов, по количеству связей и др.) позволили получить различные “графические портреты”. Так, при ранжировании по синонимам получено распределение концептов по количественным уровням синонимов (Рис.2).

Синоним концепта определяет отношение “концепт - текстовый вход концепта”. Синонимы задают текстовые входы концепта, не различимые в ситуациях их использования в текстах. Анализ гистограммы распределения концептов показывает, что максимальным числом синонимом обладают концепты СООБЩИТЬ (УВЕДОМИТЬ), ПРИМЕНЕНИЕ ОРУЖИЯ, ВЫПОЛНИТЬ (ИСПОЛНИТЬ, ОСУЩЕСТВИТЬ).

Рис.2. Гистограмма распределения синонимов концептов

Рис.3. Гистограмма распределения связей концептов

Пример синонимического ряда концепта ПРИМЕНЕНИЕ ОРУЖИЯ:
(боевое применение; ведение огня; вести огонь; нанесение удара; нанести удар; обстрел; обстреливать; обстреливаться; обстрелять; огневое воздействие; огонь; открывать огонь; открыть огонь; применение средств вооружений; применить оружие; простреливать; стрельба; стрелять; стрельнуть;).

При ранжировании по количеству связей получено распределение концептов по количественным уровням связей (рис.3). Наибольшее количество связей (рассматривается степень вершины) имеет концепт ИСТРЕБИТЕЛЬ – 30 связей. При этом состав установленных связей следующий: 23 иерархических отношения, 6 – отношений направленной ассоциации и одно отношение ЧАСТЬ-ЦЕЛОЕ. Для концепта САМОЛЕТ установлено 23 связи, из которых 10 иерархических отношений, 5 – отношений ЧАСТЬ-ЦЕЛОЕ, 8 – отношений направленной ассоциации.

Полученный результат отражает общие принципы установления отношений, существенное использование механизма наследования по транзитивным связям.

Концепты САМОЛЕТ и ИСТРЕБИТЕЛЬ связаны минимальной цепочкой иерархических отношений САМОЛЕТ - РЕАКТИВНЫЙ САМОЛЕТ -ИСТРЕБИТЕЛЬ, что позволяет концепту ИСТРЕБИТЕЛЬ наследовать свойства от концепта САМОЛЕТ, и в

свою очередь детализировать собственное описание за счет существенного введения видовых объектов (типов истребителей) по отношению НИЖЕ.

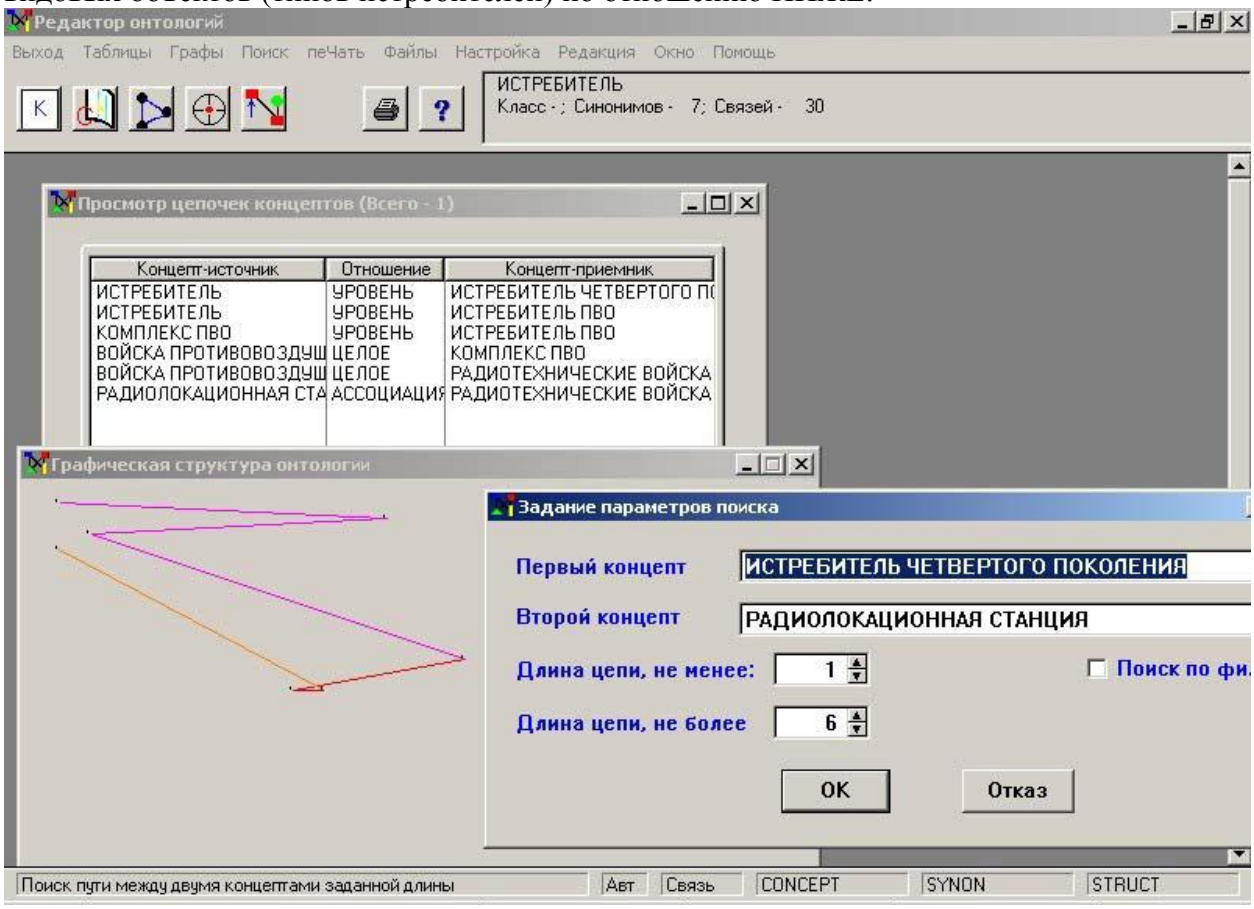

Рис.4. Поиск пути между концептами заданной длины

Особый интерес представляет анализ цепочек отношений концептов АвиаОнтологии. При построении цепочки учитывается направление связи, т.е. допустимая цепочка концептов является путем (полупуть не разрешается). Механизм построения цепочек поддерживает вывод на онтологии, который используется в прикладных задачах. При этом исследуется вопрос о структуре вывода, т.е. о числе шагов вывода, промежуточные концептах в цепочке вывода, типах связей. Очевидно, что достоверный вывод обеспечивает вывод по транзитивным связям, где не важно количество промежуточных звеньев. Структуры вывода с ассоциативными связями и вывод с перегибами по транзитивным связям требуют дополнительного изучения.

На рис. 4 приведена структура пути между концептами ИСТРЕБИТЕЛЬ ЧЕТВЕРТОГО ПОКОЛЕНИЯ и РАДИОЛОКАЦИОННАЯ СТАНЦИЯ. Минимальная длина пути – 6 связей. На форме также приведен графическая структура вывода при ранжировании концептов по номерам в базе.

Заключение

Текущая версия АвиаОнтологии представляет собой специализированный информационный ресурс, который уже на данном состоянии разработки может быть встроен в различные приложения. К числу потенциальных приложений можно отнести ряд информационно-поисковых задач, таких как содержательный поиск документов с расширением запроса по иерархии онтологии, нахождения похожих документов, классификации документов по одному или нескольким классификаторам, например, классификации по ситуациям Обороны, Атаки, Посадки, Взлета и т.п. Другие приложения связаны с задачами анализа специализированных текстов.

Развитие АвиаОнтологии предусматривает ее дальнейшую специализацию. Отдельным направлением исследований является расширение типов отношений онтологии, добавление отношений последовательностей (ситуационных и временных).

Визуальные методы проектирования онтологий способствуют более быстрому и более полному пониманию структуры знаний предметной области, что особенно ценно для исследователей, работающих в новой предметной области.

Инструментальная система «OntoEditor» предоставляет эффективный набор инструментальных средств, позволяющих осуществлять проектирование онтологии любой структуры, с любыми типами отношений и любыми классами концептов. Особенно важно, что система поддерживает импорт любого типа онтологии, реализованной на физическом уровне в табличной форме. Поддержка разнообразных поисковых запросов и механизмы вывода на онтологии позволяют исследовать внутреннюю структуру знаний предметной области. Система «OntoEditor» может быть встроена в приложения различного назначения, работающие с большими базами знаний.

Литература

1. Невзорова О.А., Федунов Б.Е., Система анализа технических текстов "ЛоТА": основные концепции и проектные решения. // Изв. РАН. Теория и системы управления. – 2001. – № 3.– С. 138-149.
2. Лукашевич Н.В., Добров Б.В., Тезаурус русского языка для автоматической обработки больших текстовых коллекций // Компьютерная лингвистика и интеллектуальные технологии: Труды Международного семинара Диалог’2002 / Под ред. А.С.Нариньяни – М.: Наука – 2002. – Т.2 - С.338-346.
3. Dobrov B., Loukachevitch N., Nevzorova O. An approach to new ontologies development: main ideas and simulation results //Int. Journal Information Theories & Applications. Vol.10. Number 1, 2003. P.98-105.
4. Guarino N., Some Ontological Principles for Designing Upper Level Lexical Resources // Proceedings of First International Conference on Language Resources and Evaluation, 1998..

Применение онтологии для автоматизации весового проектирования сложных технических объектов

А.В. ДЕРКАЧ

http://www.nbuv.gov.ua/portal/natural/Kzms/2008/2008_st18.pdf

Комп’ютерні засоби, мережі та системи. 2008, № 7 **153-159**

A.V. Derkach ***APPLICATION OF ONTOTOLOGY FOR AUTOMATION WEIGHT DEVELOPMENT OF THE COMPLEX TECHNICAL OBJECT***

Ontology using for aircraft development based on computer aid-design system is considered.
Описана концепція побудови автоматизованої системи вагового проектування літака, яка діє на початкових етапах проектування літака і підтримує правила онтології.
Описана концепция построения автоматизированной системы весового проектирования самолета, которая действует на начальных этапах проектирования самолета, и построена на основе принципов онтологии.

Введение. При проектировании любого сложного технического объекта (СТО) возникает проблема определения его весовых параметров. Особо важное значение весовое

проектирование имеет при прогнозировании весовых характеристик самолета на начальных этапах разработки его проекта. Задачами весового проектирования СТО является: оптимизация массы СТО на стадии определения его параметров и выполнения схем компоновки; минимизация массы частей СТО при разработке силовых схем и конструкций; определение характерных масс СТО и массы всех агрегатов и систем на основе весового расчета; прогнозирование окончательных значений характерных масс с учетом их изменений в процессе создания СТО; определение на основе совместных весовых и аэродинамических расчетов основных размеров и тяговооруженности; определение исходных данных для расчета СТО на прочность; весовой анализ эффективности СТО и уровня весового совершенства конструкции; выбор рационального размещения масс и др. [1 3].

Состав, алгоритмы, методики и весовые соотношения, которые применяются в расчетах при осуществлении весового проектирования, зависят от параметров конкретного изделия, сопровождаются весовым анализом, имеют огромное количество промежуточных данных и осуществляются на разных этапах проектирования СТО с разной точностью и детальностью.

В настоящее время существует очень мало программ в области весового проектирования и в основном они решают эту задачу лишь частично. Отметим, что задача весового проектирования не является чисто технической. Большую роль в весовом проектировании играют административные методы, множество сложных проблем в этой области не может быть решено без использования конструкторской интуиции и опыта. Поэтому полностью автоматизировать решение всех проблем весового проектирования невозможно. В данной работе описывается подход к построению структуры автоматизированной системы прогнозирования весовых характеристик (АСПВХ), которая автоматизирует одну из задач весового проектирования прогнозирование весовых характеристик на предпроектном этапе конструирования самолета. Система АСПВХ интеллектуально подходит к решению задач, принимает самостоятельные решения в зависимости от конкретной ситуации по многим из перечисленных ранее вопросам. Для определения масс компонентов самолета, его характерных масс существует множество различных расчетных соотношений [1 3]. Метод расчета выбирается в зависимости от этапа проектирования, типа самолета, его условий эксплуатации, геометрических и аэродинамических параметров.

Основные положения. Процесс весового проектирования представляет собой решение (в определенной последовательности) большого числа задач, связи которых основаны на взаимодополняющих исходных данных, а методика носит итерационный характер [1]. При автоматизации весового проектирования В.М. Шейнин [1] предлагает выделить 3 этапа, в соответствии с последовательностью развития проекта. Расчеты первого этапа связаны с определением внешнего облика СТО на основе данных, которые задаются техническими требованиями, или определяются по статистике. Главной задачей данного этапа является аналитическое определение весовых характеристик, размеров и параметров СТО, на основе которых выполняется первичный учет весовых параметров его основных компонентов. Второй этап расчетов связан с выбором конфигурации и размеров основных элементов СТО, определением его аэродинамических характеристик. На основе этих данных проводится более точный и детальный расчет весовых показателей компонентов СТО. Расчеты третьего этапа включают в себя много видов задач весового проектирования, в том числе определение конечных результатов весовых характеристик и количественных показателей весового проектирования (укрупненных весовых лимитов) [1]. Основные задачи системы АСПВХ на каждом этапе показаны на рис. 1. Процесс прогнозирования весовых характеристик СТО начинается с анализа факторов, которые определяют величину массы СТО. От результатов этого анализа зависит выбор методики расчета.

Процесс прогнозирования весовых характеристик происходит в условиях ограниченной информации о проектируемом изделии. Поэтому данная система должна оптимизировать свою работу с учетом имеющихся данных. С другой стороны, могут быть доступны разные характеристики СТО, которые прямо или косвенно влияют на массу: характеристики конструкционных материалов; геометрические параметры, определяющие размеры конструкций; внешние нагрузки, определяющие общую нагрузку конструкции; внутренние силы, возникающие в элементах; аэродинамические требования; основные проектные параметры; требования прочности; требования к эксплуатации (эксплуатационная технологичность, надежность, долговечность, безопасность); технологические требования (членение конструкции, требования к заготовкам, процессам обработки, соединениям); особенности организации производства СТО и т. д.

Рис. 1. Алгоритм проведения весовых расчетов в АСПВХ

Данной системе необходимо учитывать многообразие различных факторов, влияющих на вес СТО, а также всевозможные варианты методики расчета, в зависимости от типа СТО, его предназначения, аэродинамических, геометрических параметров и т. д.

Для организации хранения, обработки, систематизации большого количества разнообразной информации, установления взаимосвязей между отдельными частями системы АСПВХ, а также для поддержки принимаемых решений предложено применить онтологическую схему организации базы знаний в системе АСПВХ.

Онтологию предметной области можно представить в виде: $O = < X, R, F >$, где X – множество понятий, терминов, единиц знания; R - множество отношений между единицами знаний; F – множество функций интерпретации, правил заданных на множестве понятий и отношений [4].

Данные в области весового проектирования представляются в виде объектов, классов, связей между ними, а также правил интерпретации этих данных.

Применение онтологии позволит связать между собой такие принципиально разные понятия: компоненты самолета, массы компонентов, формулы, по которым рассчитываются эти массы, входные параметры для этих формул, а также отношения базы данных, где хранится информация. С помощью онтологии, кроме правил интерпретации и характеристики объектов, можно представить также правила выбора формул, перехода к новому состоянию процесса, условия проведения анализа и правила определения предыдущего состояния процесса. Следовательно, система АСПВХ получает возможность повторить или изменить историю расчета масс компонентов самолета.

Основными понятиями онтологии весового проектирования является самолет, компоненты самолета, массы компонентов самолета, особенности компоновки самолета, методы расчета, функции расчета, модули (структурные единицы системы), отношения в базе данных, пользователи системы, геометрические, аэродинамические и другие характеристики самолета или его компонентов и условия применения методов расчета. Взаимодействие данных понятий показано на рис. 2.

Полученная система классов, подклассов, атрибутов и связей, реализованная на языке OWL [5, 6], а также функции интерпретации и правила выбора, реализованные на языке Java, составляют основу базы знаний и системы принятия решений АСПВХ. Онтология в данном случае связывает между собой отдельные компоненты системы, формирует необходимые данные для принятия решений, управляет ходом расчетов и весовым анализом, создает историю проекта и обеспечивает обратную связь между отдельными его состояниями.

Таким образом, система АСПВХ представляет собой совокупность отдельных модулей, которые объединяются между собой с помощью онтологических схем. Каждая функция расчета помещается в отдельный модуль, что в целом образует библиотеку компонентов. С помощью правил онтологии определяется компонент библиотеки, который должен быть применен на определенном этапе весового расчета, проводится анализ необходимых данных, определяется место хранения параметров расчета, проверяется полнота данных, организуется и оформляется запрос дополнительных данных у пользователя. Структура основных компонентов системы показана на рис. 3.

Обычно онтология применяется для организации трансляции текста с естественного языка на формальный, поиска информации по нечетко определенным запросам, формирования словаря определенной области данных или обслуживания интерфейсов. Использование ее для объединения программных частей как основы принятия решений и формирования процесса вычисления является не менее эффективным.

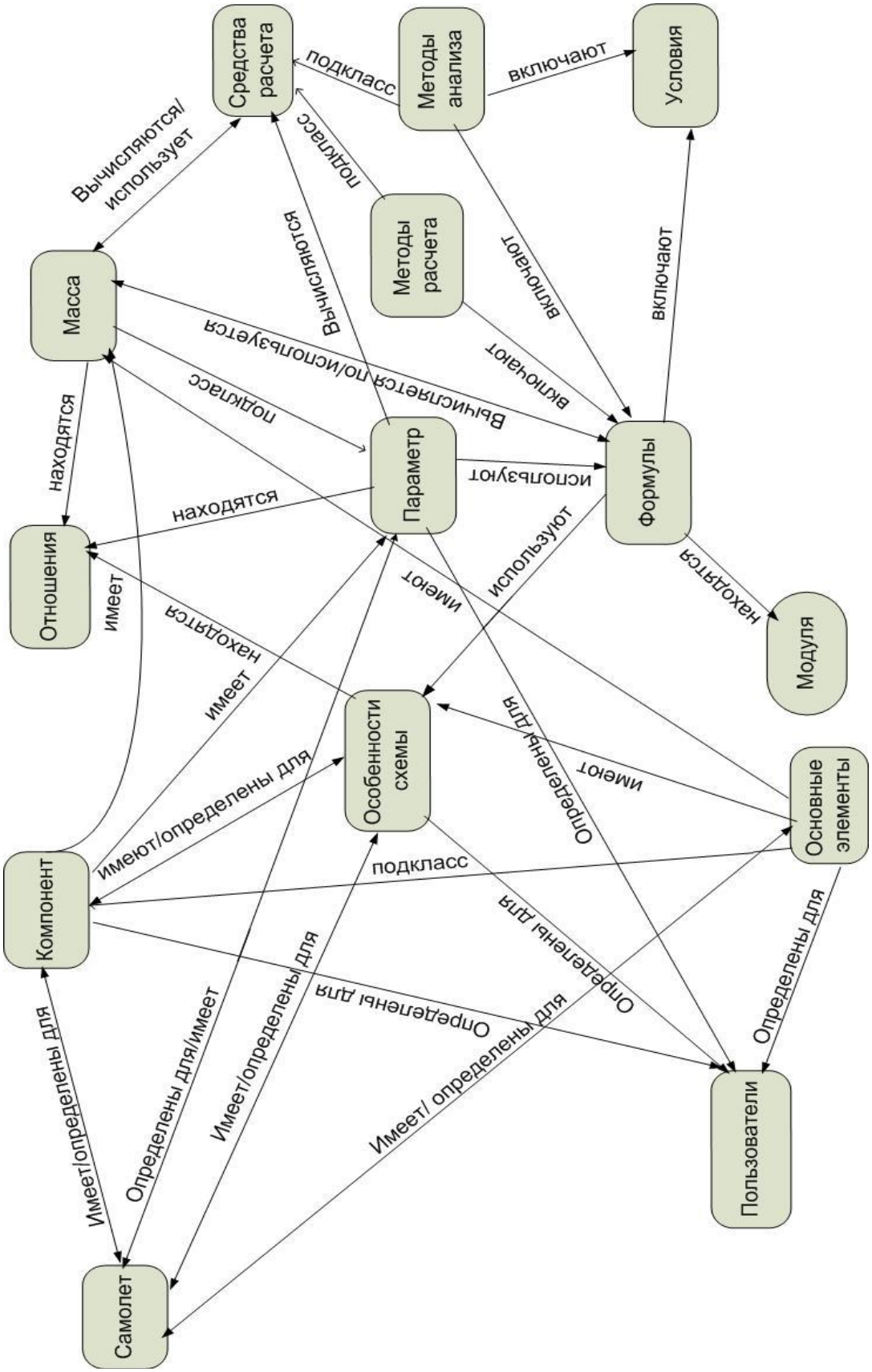

Рис.2. Схема связей классов в заданной предметной области

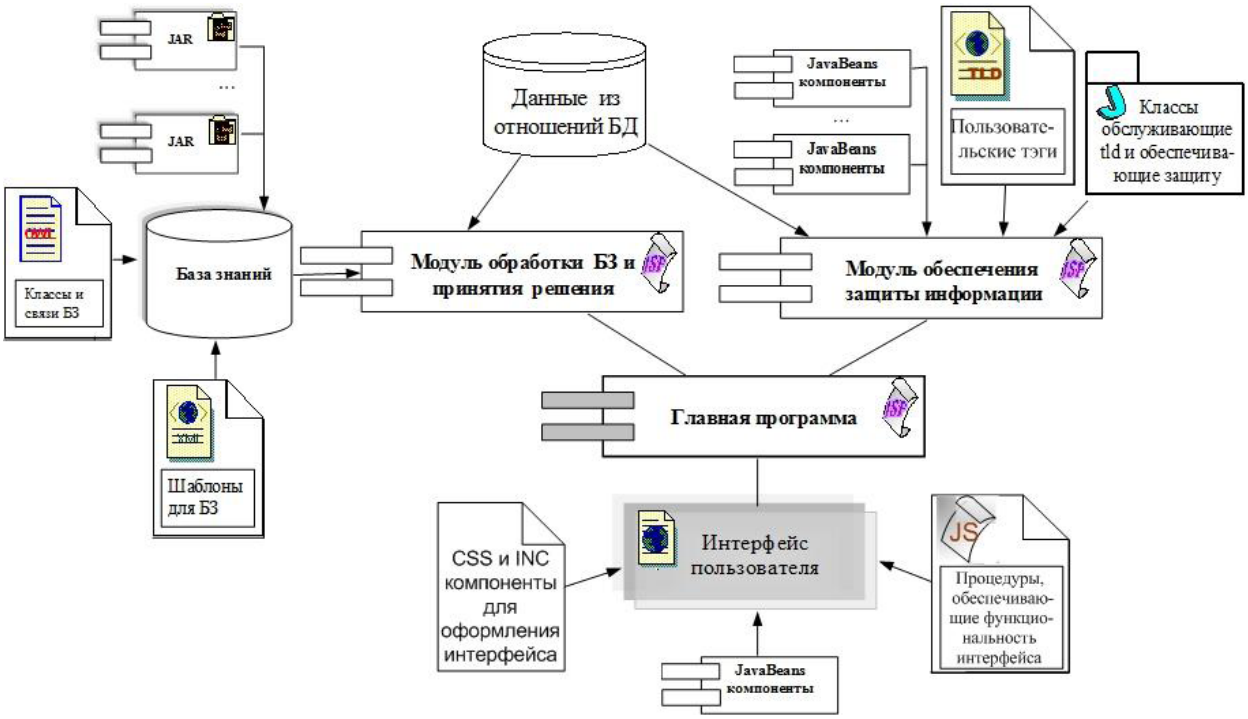

Рис.3. Структура основных компонентов АСПВХ

Выводы.

1. Система АСПВХ обеспечивает комплексное решение одной из важней-ших проблем весового проектирования – задачи прогнозирования характеристик самолетов и их модификаций на предпроектном этапе создания самолета.

2. В основе построения системы АСПВХ лежит онтологический подход. Благодаря этому обеспечивается взаимосвязь структурных частей в составе АСПВХ, т. е. система получает знания о своих компонентах, анализируя имеющиеся в библиотеке программные модули и выбирая оптимальный модуль для решения задачи в конкретных условиях.

3. Созданная база знаний системы АСПВХ может легко наращивать- ся, включая новые типы СТО, методы и средства весовых расчетов, оценку влияния на массу СТО новых материалов и схем конструкции. Это позволяет системе АСПВХ легко адаптироваться к быстро изменяющимся технологиям создания СТО.

1. *Шейнин В.М., Козловский В.И*. Весовое проектирование и эффективность пассажирских самолетов: Справочник. – 2-е изд., перераб. и доп. – М.: Машиностроение, 1984. – 552 с.
2. *Егер С.М., Мишин В.Ф., Лисейцев Н.К. и др*. Проектирование самолетов: Ученик для вузов / Под ред. С.М. Егера. – 3-е изд., перераб. и доп. – М.: Машиностроение, 1983. – 616с.
3. *Бадягин А.А, Егер С.М., Мишин В.Ф. и др.* Проектирование самолетов. – М.: Машиностроение, 1972. – 516 с.
4. *Палагин А.В., Яковлев Ю.С.* Системная интеграция средств компьютерной техники. Монография. Винница: УНИВЕРСУМ-Винница, 2005. – 680 с.
5. Asuncion Gomez-Perez, Mariano Fernandez-Lopez, Oscar Corcho. Ontological Engineering, Springer-Verlag London Limited, 2004. – 411 p.
6. *Nicola Guarino.* Formal Ontology and Information Systems, National Research Council, LADSEB-CNR, Padova, Italy, P. 3–15.

Онтологическая поисковая система Jewel для реализации интеллектуального поиска в Интернет- и интранет-сетях

Электронный журнал Труды МАИ

Е.В. Сизиков, Д.В. Сошников

В работе рассказывается о применении искусственного интеллекта к решению задачи поиска в Интернет- и интранет-сетях с использованием онтологий. Вводится в рассмотрение онтологический подход к аннотированию ресурсов, предлагающий внедрение в веб-страницы дополнительной структурированной информации об их содержании, которая в дальнейшем используется поисковым роботом для построения фреймовой индексной базы знаний, на основе которой производится определение релевантности найденного ресурса запросу. Также обсуждаются особенности реализации прототипа поисковой системы на основе предложенного подхода.

Искусственный интеллект – одно из интереснейших направлений научной деятельности человечества, которое ставит перед собой задачу построения вычислительного устройства, способного, подобно человеку, к самостоятельному мышлению. Методы искусственного интеллекта, как правило, используются для решения неформальных задач, постановка которых проста и понятна для человека, но для которых сложно указать алгоритм их решения.

В настоящее время характерна тенденция к использованию элементов искусственного интеллекта для решения проблемы поиска информации. За последние годы представлено большое число работ на тему интеллектуального поиска. Интерес к данной теме вполне оправдан как с научной, так и с практической точки зрения. Развитие методов интеллектуального поиска с одной стороны позволяет приблизиться к решению задачи автоматизированной структуризации больших массивов слабоструктурированных данных; кроме того, привнесение интеллектуальности в обычно рутинный процесс поиска информации в существующих системах может перевернуть индустрию поисковых систем и вывести качество обслуживания в данной области на совершенно новый уровень.

Существует множество различных подходов к использованию технологий искусственного интеллекта для решения задачи поиска. Одним из возможных путей может стать аннотирование Web страниц дополнительной структурированной информацией об их содержании, которая в дальнейшем может быть использована поисковой машиной для определения релевантности запросу найденного Internet ресурса. В данной работе рассматривается **онтологический подход** к аннотированию веб-ресурсов.

Онтология – учение о познании (от греч. онтос – сущее, логос - понятие). В инженерии знаний под онтологией понимается детальное описание некоторой предметной области, которое используется для формального и декларативного определения ее концептуализации. Зачастую онтологией называют базу знаний специального вида, которую можно разделять, отчуждать, и самостоятельно использовать в рамках рассматриваемой предметной области [1].

Онтологические системы могут применяться для решения различных задач в сфере искусственного интеллекта, но, пожалуй, наиболее характерной сферой их применения является представление знаний в Интернет. Круг связанных с этим вопросов весьма широк и включает в себя мультиагентные системы, автоматическое извлечение знаний из

текстов на естественном языке, поиск информации, интеллектуальное аннотирование, автоматическое составление авторефератов и проч.

Примером общей онтологической системы является CYC, разрабатываемой фирмой CYCorp [2]. Проект включает в себя создание обширной онтологической системы, описывающей более чем 10^6 концептов и 10^5 аксиом. Для представления знаний фирма разработала специальный язык CYCL. Для вывода по онтологической базе знаний разработана специальная машина вывода. Основная цель этого проекта — построение обширной базы знаний обо всех общих понятиях практически во всех областях человеческой деятельности (common knowledge).

Другим примером использования онтологической системы является инициатива (KA)2 [3] (Knowledge Annotation Initiative of the Knowledge Acquisition Community). Это международный проект, целью которого является организация интеллектуального поиска в Интернет и автоматическое накопление новых знаний. В этой инициативе выделяют следующие направления:

Аннотация web страниц интеллектуальной информацией.

Онтологический инжиниринг.

Организация интерфейса запросов и вывода по распределенной онтологии.

Аннотация Web страниц осуществляется за счет расширения HTML специальным тегом <ONTO>, в рамках которого можно задавать онтологии для спецификации WWW страниц. Процесс дополнения Web страниц такой онтологической информацией рассматривается в рамках направления онтологического инжиниринга. В онтологический инжиниринг входит разработка собственной онтологической системы на основе инструментария Ontolingua. На настоящий момент разработано более десятка онтологий для описания организаций, направлений исследований, рабочих процессов, личностей, продуктов производства и пр. Для поиска в рамках (KA)2 предполагается использовать подсистему Ontocrawler, а для организации запросов и вывода разработан программный продукт Ontobroker, со своим внутренним языком запросов. Существует проекты меньшего масштаба, к ним можно отнести, например, перспективную систему SHOE, разрабатываемую кафедрой информатики университета в Мериленд (Department of computer Science of Maryland University)[4].

Общим для всех систем онтологического аннотирования является то, что в качестве аннотации веб-ресурса выступает специальным образом организованная предметная онтология, которая содержит структурированные знания об аннотированном ресурсе относительно некоторой метаонтологии предметной области. Можно предложить различные способы размещения онтологической информации о ресурсе: включить онтологическое описание в HTML код через введение новых HTML тегов, либо хранить онтологическое описание ресурса в отдельном файле в каком-либо специальном представлении.

Основная задача онтологического подхода состоит в том, чтобы облегчить пользователю поиск информации в большом наборе ресурсов за счет систематизации знаний, создания единой иерархии понятий, унификации терминов и правил интерпретации. Для описания онтологий можно использовать различные языки представления знаний, применяемые, например, в экспертных системах. В данной работе предлагается использовать для составления онтологических описаний фреймово-продукционный способ представления знаний.

Как известно, **фреймы** [5] — средство описания статических знаний, удобное для описания иерархии абстрактных и конкретных понятий, близкое к объектно-ориентированному подходу [6]. Продукции, определенные над множеством фреймов и их слотов, позволяют описывать динамические знания.

В тоже время, применение фреймово-продукционных языков представления знаний в "чистом" виде недостаточно для организации эффективного онтологического поиска. Это видно, например, из следующего соображения.

Пусть всякое онтологическое описание внедрено только в описываемый этой онтологией ресурс. Мы будем считать ресурс известным, если мы в любой момент имеем доступ к его содержимому и, как следствие, к онтологическому описанию. Предположим, что нам уже известно некоторое множество онтологий, и мы хотим указать поисковой системе, какие еще онтологии мы хотели бы сделать известными. Для этого необходимо указать некоторое правило - поисковый запрос, который отделит искомые онтологии от всех остальных, имеющихся в системе. Однако, в общем случае, отсутствует возможность узнать, какие знания содержатся в онтологии до того, как она стала известной.

Таким образом, для поискового запроса не существует никаких явных связей между онтологиями в фреймово-продукционном представлении, кроме отношения наследования между их фреймами (Другие связи определены продукционными правилами. Они неизвестны до тех пор, пока не стала известной сама онтология). Очевидно, остается только возможность сделать запрос следующего типа: "найти все онтологии, фреймы которых унаследованы от данного известного фрейма и значения слотов которых удовлетворяет некоторому условию".

Как видно, запрос состоит из условий, накладываемых на отношение наследования и на значения слотов наследников. Но в тоже время существует опасность, что слот наследника изменил свой первоначальный смысл, так как в общем случае это уже другой фрейм, который может иметь произвольную структуру.

В данной работе предлагается модифицировать фреймовое представление знаний, явно разделив фреймы-образцы и фреймы-экземпляры, введя требование запретить изменять структуру или применять наследование к фреймам-экземплярам. В дальнейшем мы будем называть **категорией** фрейм-образец, а под **концептом** будем понимать фрейм-экземпляр. Категория во всем эквивалентна обычному фрейму, кроме того, что значения ее слотов воспринимаются концептами как значения по умолчанию, а концепт соответственно является точной копией своей категории с точностью до значений слотов и безусловных правил, явно присваивающих слоту его значение.

Внедрение подобного подхода позволяет существенно обогатить множество возможных поисковых запросов. Действительно, пусть существует некоторая предметная область и некоторое множество текстовых ресурсов, ее описывающих. Если сосредоточить фактические описания явлений и закономерностей - то есть категорий предметной области в нескольких онтологиях страниц, то появляется возможность искать нужную информацию во множестве страниц посредством поиска онтологий, концепты которых соответствуют требуемым условиям. Теперь условия запроса могут касаться как отношений наследования между категориями или отношений представления между категориями и их концептами, так и условий, накладываемых на значения слотов для концептов известных категорий. В сравнении с предыдущим примером имеется гарантия, что наследники не претерпели никаких метаморфоз, так как концепт нельзя дополнить новыми слотами или продукциями.

Таким образом, открывается возможность разделить поиск информации на два этапа: вначале изучается описание существующих явлений, а затем ведется поиск частных случаев изученных явлений. Это обстоятельство, при условии уникальности используемых имен, дает дополнительное преимущество, состоящее в том, что вводится принудительная унификация понятий в рамках одной предметной области, что исключает возможность двусмысленности поискового запроса.

Язык составления онтологических описаний

Для составления онтологических описаний в рамках создания онтологической поисковой системы Jewel была проведена разработка общего языка описания онтологий. В основе предлагаемого языка лежит фреймово-продукционный язык JFMDL из состава инструментария JULIA (Java Universal Library for Intelligent Applications) [7,8,9], расширенный согласно вышеописанным положениям.

Язык позволяет производить онтологические описания HTML страниц, используя понятия: категория, условное правило, безусловное правило и концепт. Под онтологией HTML страницы (онтологией части предметной области, описываемой в странице) понимается описание некоторого ресурса, проводимое в терминах общего языка описания онтологий.

В целях повышения эффективности поиска онтологий и непротиворечивости их описания принимаются следующие соглашения:

- Каждая онтология HTML страницы предназначена для непосредственного описания той страницы, на которой она находится. Причем в теле страницы может быть определена только одна онтология.
- Каждая онтология обладает набором предопределенных свойств:
  - именем, которое совпадает с физическим местоположением HTML страницы, в теле которой содержится описание онтологии;
  - списком используемых онтологий (для описания категорий и правил создаваемой онтологии могут применяться категории и правила объявленных используемых онтологий) и их внутренних имен, ассоциированных с ними для удобства;
  - кратким словесным описанием.

Для описания онтологии используется надмножество стандарта HTML, в котором расширяется стандартный тег <SCRIPT>, а также вводятся новые теги <USE>, <CONCEPT>, <SET>, <ASSIGN>. Рассмотрим теперь подробнее теги, используемые в описании онтологий.

Приведем простой пример онтологического описания некоторой предметной области. В качестве предметной области рассмотрим справочник по моделям самолетов, представленный набором HTML страниц — по одной на каждую модель. Мы можем выделить пассажирские и транспортные самолеты. Объединим эти сведения в главной странице - aircrafts.html.

Страница aircrafts.html

```
. . .
<SCRIPT language = ONTODEF>
CATEGORY Firm
{
SCALAR name;
SCALAR country;
}
CONCEPT Ilushin IMPLEMENTS Firm;
SET Ilushin .name = 'Il';
SET Ilushin .country = 'Russia';
CONCEPT Tupolev IMPLEMENTS Firm;
SET Tupolev .name = 'Tu';
SET Tupolev.country = 'Russia';
CATEGORY Plane
{
SCALAR name DEF 'Plane'; // Название самолета
LIST modifications DEF []; // Список возможных модификаций
REF firm; // Указатель на концепт, описывающий производителя
SCALAR type; // Тип самолета (сверхзвуковой/дозвуковой)
SCALAR speed; //Скорость самолета
}
IF Plane.speed<=1250 THEN Plane.type ='subsonic';
IF Plane.speed>1250 THEN Plane.type ='supersonic';
SET Plane.type = 'speed is unknown';
CATEGORY PassengerPlane EXTENDS Plane
{
SCALAR passengers; // Число пассажиров
```

```
}
CATEGORY TransportPlane EXTENDS Plane
{
SCALAR mass; // Масса полезной нагрузки
}
</SCRIPT>
. . .
```

Теперь любая страница, содержащая информацию о конкретном самолете, может быть дополнена онтологическим описанием, например, следующим образом:

Страница tu-154.html

```
. . .
<USE 'aircrafts.html' AS aircraft >
<CONCEPT tu154 IMPLEMENTS @ aircraft ~PassengerPlane>
<ASSIGN tu154.name> Tu-154 </ASSIGN>
<SET tu154. firm = @Tupolev>
<SET tu154.speed = 900>
<SET tu154.modifications = $['Tu-154A', 'Tu-154M']>
<SET tu154.passengers = 100>
. . .
```

Таким образом, создается возможность для организации предметной онтологии, состоящей из некоторого числа онтологий HTML страниц.

Язык поисковых запросов

Для составления поисковых запросов в системе Jewel применяется специализированный язык, состоящий из следующего набора операторов:

Оператор **SEARCH** имеет следующую форму:

```
SEARCH
USE 'адрес_1' AS имя_1
. . .
USE 'адрес_N' AS имя_N
IMPORT LIBRARY имя_библиотеки_1
. . .
IMPORT LIBRARY имя_библиотеки_M
WHERE "условие"
```

Под условием понимается логическое выражение, определяющее искомые онтологии. В процессе поиска производится обход всех подходящих запросу онтологий (для повышения быстродействия все проверяемые онтологии, при помощи индекса, предварительно отбираются в кандидатное множество), и к элементам каждой из них применяется указанное поисковое условие. В качестве результата возвращаются онтологии, для которых условие истинно.

Для задания условия могут использоваться следующие предикаты:

- **INHERITED** (имя_категории) - принимает истинное значение в текущей онтологии, если имеется категория, унаследованная непосредственно от указанной в аргументе. В противном случае предикат принимает ложное значение.
- **EXTENDS** (имя_категории) - принимает истинное значение в текущей онтологии, если имеется категория, унаследованная (возможно не непосредственно) от указанной в аргументе. В противном случае предикат принимает ложное значение.
- **IMPLEMENTS** (имя_категории) - принимает истинное значение в текущей онтологии, если имеется концепт, представленный категорией, указанной в аргументе. В противном случае предикат принимает ложное значение.

Кроме предикатов в условие входят так называемые неявные выражения над концептами. Так, например, выражение (имя_категории.имя_слота >"значение") означает, что выражение будет истинно в случае, если текущая онтология имеет концепт указанной категории, и выражение для его слота истинно (для приведенного примера это означает, что значение, хранимое в слоте концепта, должно быть больше указанного).

Для проверки истинности выражения, при помощи обратного логического вывода, производится вычисление значения слота и последующее сравнение. В случае, если значение слота не вычислимо — выражение признается ложным.

Все выражения и предикаты в условии запроса могут быть связаны логическими операциями AND, OR и NOT.

Оператор EXTRACT имеет следующие три формы:
• EXTRACT BASE - возвращает адреса всех зарегистрированных в системе онтологий;
• EXTRACT ROOT - возвращает адреса всех зарегистрированных в системе онтологий, которые не используют никаких других онтологий;
• EXTRACT ONTOLOGY 'адрес' - возвращает онтологическое описание страницы, зарегистрированной по указанному адресу.

Рассмотрим более подробно процесс поиска информации в предлагаемой поисковой системе. Допустим, что имеется некоторая предметная область, для которой составлены все необходимые онтологические описания. Ставится задача найти страницу, в тексте которой описан некоторый факт. В терминах, введенных в данной работе, для описания явлений используется понятие категории, а для указания частных случаев явлений — концепты. Таким образом, требуется найти страницу, онтология которой содержит концепт некоторой неизвестной категории. Как видно, в общем случае, вначале требуется найти категорию, описывающую нужное явление. Затем требуется отыскать концепт найденной категории, описывающий требуемый факт. Онтология, содержащая найденный концепт, будет онтологией искомой страницы. Общий алгоритм поиска для предлагаемой поисковой системы будет сводиться к следующим действиям:

- Определение корня онтологий - именно с коренных онтологий можно начать изучение структуры онтологических описаний в случае, если структура введенной в рассмотрение предметной онтологии неизвестна. Изучение онтологий найденных страниц проводится посредством просмотра с помощью команды EXTRACT ONTOLOGY.
- Изучение описаний известных явлений предметной области до тех пор, пока не будет найдена категория, концепт которой может оказаться искомым фактом. При этом поиск новых онтологий ведется преимущественно с применением предикатов типа IMPLEMENTS, INHERITED и EXTENDS к известным категориям.
- Определение отличительных особенностей искомого концепта и непосредственный поиск концепта исходя из его отличительных особенностей. Поиск онтологии, очевидно, должен вестись с использованием неявных выражений над категориями.

Приведенный алгоритм легко продемонстрировать на ранее приведенном примере. Выделение коренных онтологий командой EXTRACT ROOT даст в качестве результата адрес онтологии страницы aircrafts.html, так как она не использует в своем описании других онтологий. Страницу самолета Ту-154 легко можно найти по названию самолета:

SEARCH
USE ' aircrafts.html' **AS** aircrafts
WHERE (@aircrafts~Plane.name == 'Tu-154') **AND** (**IMPLEMENTS**(PassengerPlane))

Или зная, например, что искомый самолет дозвуковой и берет на борт до 100 человек:

SEARCH

USE ' aircrafts.html' **AS** aircraft
WHERE (@aircraft~Plane.type = = 'subsonic')
AND(@aircraft~PassengerPlane.passengers = = 100)

Последний из вышеприведенных примеров наглядно показывает элемент интеллектуальности проводимого поиска, так как информация о том, что самолет Ту-154 дозвуковой, явно нигде не указывалась, а была выведена логически по продукционному правилу, общему для всех концептов, прямо или косвенно представляющих категорию Plane.

Легко заметить, что быстрота и качество поиска существенно зависят от качества составления онтологических описаний. Для предметной онтологии, в которой категории разбросаны по слишком большому числу онтологий страниц, поиск затруднен. Однако очевидно, что для большого объема текстов было бы неправильно сосредоточить все категории в одной онтологии, как это было сделано в вышеприведенном примере. Такой шаг может привести к нарушению смыслового разделения между понятиями и повредить точности и выразительности онтологического описания. Это, в свою очередь, негативно скажется на времени поиска, так как пользователь будет вынужден работать с большим множеством получаемых в качестве ответов страниц, аналогично тому, как это происходит при поиске по ключевым словам в классических поисковых машинах. Для более сложных онтологических систем характерно присутствие трех логических уровней:

- Первый уровень - это уровень общих абстракций. Этот слой онтологических описаний объединяет в себе все понятия предметной области и одновременно не проводит никакой конкретизации понятий.
- Второй уровень - уровень описания явлений. В этой части онтологического описания указываются конкретные явления, максимально приближенные к реальности.
- Третий уровень - предметных концептов, т.е. реализация явлений, описанных во втором уровне.

Для небольших онтологических описаний возможно сращивание первого и второго уровня, как в приведенном примере.

Как легко видеть, в таком случае поиск состоит из двух взаимосвязанных частей: поиск описаний явлений и поиск конкретных реализаций этих явлений.

Вопросы реализации

Реализация опытного прототипа системы онтологического поиска Jewel производилась на языке Java. В основу реализации был положен инструментарий JULIA для создания распределенных интеллектуальных систем на основе продукционно-фреймового представления знаний. С использованием технологии JavaCC[13] были разработаны трансляторы с языка онтологического описания веб-ресурса во внутреннее представление JULIA, а также интерпретатор языка поисковых запросов.

В процессе работы системы онтологические описания вручную (при помощи специальных команд языка запросов) или автоматически (при помощи автономного робота) транслируются во внутреннее представление, которое затем сохраняется в виде семейства индексных файлов или в объектной базе данных. Таким образом, множество известных систем онтологий проиндексировано и сохраняется на поисковом сервере в виде множества фрейм-миров.

Для реализации пользовательского интерфейса реализованы утилита администрирования поисковой системы и предоставляющий пользователю возможность формулировать поисковые запросы Java-сервлет. В качестве продолжения работы над проектом предполагается разработка более удобного диалогового интерфейса с возможностью просмотра множества известных системе онтологий категорий и концептов.

Перспективы применения и развития

В процессе работы над системой Jewel выявились некоторые возможные пути дальнейшего совершенствования методики онтологического поиска.

Рассмотрим улучшения, которым можно подвергнуть язык запросов. Прежде всего, следует отметить, что возможности языка поисковых запросов могут быть существенно расширены за счет введения возможности поиска не только онтологий, но и их составляющих - категорий и концептов. Это позволит создавать вложенные поисковые запросы, увеличив тем самым выразительность языка.

Кроме того, весьма полезным может оказаться введение в язык возможности поиска множества возможных значений для атрибутов категорий, которые принимаются в ее концептах, поскольку частой является ситуация, когда пользователю точно неизвестны нужные значения слотов для выделения искомых концептов из всех существующих.

Немаловажные изменения можно внести в язык составления онтологических описаний. Очевидным недостатком разработанной системы является отсутствие ее способности к обучению, поэтому полезным может оказаться добавление функций, позволяющих динамически, на этапе выполнения логического вывода, самостоятельно генерировать новые категории и концепты, а также включать их в онтологические описания. Данная функциональная особенность важна для придания системе способности адаптироваться к запросам пользователя и подстраиваться под его интерпретацию онтологического описания, исходя, например, из сопоставления множеств запросов и найденных по ним страниц.

Безусловно, для придания системе возможностей самостоятельного совершенствования необходима, прежде всего, обратная связь, дающая системе материалы для анализа результатов своих ответов.

Разработка методов самообучения для поисковой машины может привести к созданию принципиально нового поколения поисковых систем, онтологические описания ресурсов которых совершенствуются в жизненном цикле такой системы без прямого участия человека.

Нельзя обойти вниманием и уже наметившуюся тенденцию к созданию программных средств автоматической генерации онтологических описаний. В основе данных разработок лежит анализ естественного языка. К сожалению, особенно революционных достижений в данной области обнаружить не удается, но наметилась объективная тенденция к росту возможностей таких систем.

Прогресс в области естественно-языкового анализа в будущем неизбежно затронет и языки поисковых запросов, что будет приближать разработчиков к созданию более интеллектуальных поисковых систем.

Список литературы

1 Гаврилова Т. А., Хорошевский В. Ф. Базы знаний интеллектуальных систем. - С-Пб.: Питер, 2000. - 384 с.

2 http://www.cycorp.com (02.10.2001)

3 www.ksl.svc.stanford.edu (02.10.2001)

4 Luke S., Heflin J. *SHOE 1.0 Proposed Specificatio*n. - http://www.cs.umd.edu/projects/plus/SHOE (28.09.2001)

5 Минский М. Фреймы для представления знаний. -М.: Энергия, 1979 - 342 с.

6 Буч Г. Объектно-ориентированное проектирование с примерами применения. -М.: Мир, 1992. - 286 с.

7 Soshnikov D. Software Toolkit for Building Embedded and Distributed Knowledge-Based Systems. Proceedings of the 2nd International Workshop on Computer Science and Information Technologies, Ufa, 2000. - pp. 103 - 111.

8 Soshnikov D. Technologies for Building Intelligent web applications based on JULIA Toolkit In Proceedings of the 3rd International Workshop on Computer Science and Information Technologies, Ufa, 2001. - pp. 23-34
9 Сошников Д. В. Инструментарий JULIA для создания распределенных интеллектуальных систем на основе фреймово-продукционного представления знаний. // Труды МАИ. - 2002, № .
10 Gruber. T. Towards Principles for the Design of Ontologies used for Knowledge Sharing // International Journal of Human and Computer Studies. - 1995, №43(5/6). - pp. 907-922.
11 Stoffel K., Taylor M., Hendler J. Efficient management of very large ontologies. *Proc. of Fourteenth American Association for Artificial Intelligence Conference (AAAI-97*), Menlo Park, CA, AAAI/MIT Press.Villemin F. - 1997 - pp. 12-21.
12 Brachman R., Levesque H. The tractability of subsumption in frame-based description languages. *Proc. of the National Conference on Artificial Intelligence (AAAI-1984*), Menlo Park, CA, AAAI/MIT Press - 1984. - pp. 34–37.
13 http://www.webgain.com/java_cc (13.10.2001)

Прикладная онтология на языке гиперграфов

II Всероссийская конференция «ЗНАНИЯ – ОНТОЛОГИИ – ТЕОРИИ»
с международным участием 20–22 октября 2009 г. Новосибирск

Г.К. Хахалин
Научно-Исследовательский Центр Электронной Вычислительной Техники, Варшавское шоссе, д. 125, г. Москва, 113405, Россия.
gkhakhalin@yandex.ru

Аннотация. *В данной работе рассматривается гиперграфовый язык представления прикладной онтологии. Прикладная онтология рассматривается в качестве единого концептуального интерфейса между подсистемами интегральной системы искусственного интеллекта.*
Ключевые слова: гиперграф, язык представления знаний, прикладная онтология, концептуальный интерфейс

1 Введение

В последнее время в искусственном интеллекте наметилась тенденция интегрирования различных подсистем в единую систему. Для этого придуман термин Artificial General Intelligence, что можно было бы перевести как Интегральный Искусственный Интеллект (ИИИ). В 2008 году была проведена в Мемфисе (США) первая международная конференция по этой тематике [1], в 2009 – намечена вторая.

В рамках аналогичного проекта предполагается интеграция в ИИИ разномодальных систем сенсорного, моторного и ментального характера [2]. На первом этапе - подсистем анализа и синтеза изображений, анализа и синтеза естественного языка и подсистемы решения задач с дальнейшим расширением функций анализа и синтеза речи, ответов на вопросы, объяснений собственного поведения, обучения и самообучения и т.д.

Совершенно очевидно, что интеграция разномодальных систем должна осуществляться при наличии единого концептуального интерфейса между ними. В качестве такого интерфейса предлагается онтология – концептуальная «модель мира», в которой «работает» ИИИ. В этом случае некоторые из подсистем необходимо «доводить»

до концептуального уровня, поскольку степень такой проработки у подсистем на сегодняшний день различна.

Если в качестве единого интерфейса берется онтология, то возникает два вопроса. Выбор языка представления онтологических знаний, чтобы он в дальнейшем позволил «погружать» в него требуемые расширения, например, нечеткие знания, познавательные процедуры и т.д. Второй вопрос относится к разработке прикладной онтологии общей для подсистем, способной на концептуальном уровне интегрировать разномодальные входы и выходы ИИИ. При этом, конечно, предполагается, что «внутри» каждой подсистемы могут быть свои языки представления и свои базы «внутренних» знаний. Например, для зрительной системы анализа изображений существует своя база для выделения «непроизводных» признаков; для системы анализа текста – своя для морфологии и синтаксиса естественного языка.

В данной работе рассматривается гиперграфовый язык представления концептуальных знаний и прикладная онтология «Планиметрия». Онтология «Планиметрия» выбрана из-за возможности отрабатывать на ней зрительное восприятие, синтез зрительных объектов, анализ и синтез текстов, а также планирование решения планиметрических задач. В дальнейшем при наличии ядра этой прикладной онтологии можно наращивать эту систему блоками «ответов на вопросы», «объяснения» и привлекать познавательные процедуры (индукцию, аналогию и др.), а также ставить задачу обучения и самообучения как развитие известных методов, приближаясь к «человеческому» обучению. Наличие воображения позволяет экстраполировать данную онтологию до практически значимых моделей знаний, где комплексно решается более широкий перечень интеллектуальных задач.

2 Язык представления знаний – семантический гиперграф

Для разных предметных областей и для разных задач существует спектр языков (моделей) представления знаний. Обзоры некоторых из них даны в [3]. На наш взгляд для системы ИИИ, охарактеризованной выше, наиболее адекватным языком представления знаний является язык гиперграфов в качестве расширения семантических сетей.

В общем случае гиперграф можно задать различными способами. Определим гиперграф согласно [4, 5]. Гиперграф $\mathbf{H}$ ($\boldsymbol{V}$, $\boldsymbol{E}$) есть пара, где $\boldsymbol{V}$ – множество вершин $\boldsymbol{V} = \{v\mathrm{i}\}$, $\mathrm{i} \in \mathrm{I} = \{1, 2,\ldots, n\}$, а $\boldsymbol{E}$ – множество ребер $\boldsymbol{E} = \{e\mathrm{j}\}$, $\mathrm{j} \in \mathrm{J} = \{1, 2,..., m\}$; каждое ребро представляет собой подмножество $\boldsymbol{V}$. Вершина v и ребро e называются *инцидентными*, если $v \in e$. Для $v \in \boldsymbol{V}$ через $d(v)$ обозначается число ребер, инцидентных вершине v; $d(v)$ называется *степенью вершины* v. *Степень ребра* e – число вершин инцидентных этому ребру, - обозначается через $r(e)$. Гиперграф $\mathbf{H}$ является *r-однородным*, если все его ребра имеют одинаковую степень r. Если каждое его ребро имеет степень равную 2, то гиперграф является графом.

Гиперграф ($\boldsymbol{V}'$, $\boldsymbol{E}'$) называется *подгиперграфом* ($\boldsymbol{V}$, $\boldsymbol{E}$), если $\boldsymbol{V}' \subseteq \boldsymbol{V}$, $\boldsymbol{E}' \subseteq \boldsymbol{E}$ и вершина $v \in \boldsymbol{V}'$ и ребро $e \in \boldsymbol{E}'$ инцидентны в ($\boldsymbol{V}'$, $\boldsymbol{E}'$) тогда, и только тогда, когда они инцидентны в ($\boldsymbol{V}$, $\boldsymbol{E}$).

Гиперграфы бывают *ориентированные* и *неориентированные*. Рёбра неориентированного гиперграфа называются *звеньями*. В случае ориентированного гиперграфа (оргиперграфа) ребро $e \in \boldsymbol{E}$ называется *гипердугой* (для краткости *дугой*) и представляется как упорядоченная пара (h, $\boldsymbol{T}$), где $h \in \boldsymbol{V}$, $\boldsymbol{T} \subseteq \boldsymbol{V}\backslash\{h\}$, $\boldsymbol{T} \neq \varnothing$. При этом вершина h называется *началом* дуги e, а каждая вершина из $\boldsymbol{T}$ — *конечной вершиной* дуги e. Будем говорить, что дуга e *исходит* из вершины h и *заходит* в каждую из вершин множества $\boldsymbol{T}$.

Введем понятие частично ориентированного гиперграфа. Гиперграф $\mathbf{H}$ = ($\boldsymbol{V}$, $\boldsymbol{E}1$, $\boldsymbol{E}2$) называется *частично ориентированным* или *квази ориентированным гиперграфом*, если $\boldsymbol{E}1 \subset \boldsymbol{E}$ & $\boldsymbol{E}1 \neq \varnothing$ - множество ребер (звеньев), а $\boldsymbol{E}2 \subset \boldsymbol{E}$ & $\boldsymbol{E}2 \neq \varnothing$ - множество дуг и $\boldsymbol{E}1 \cup \boldsymbol{E}2 = \boldsymbol{E}$. Если $\boldsymbol{E}1 = \boldsymbol{E}$ & $\boldsymbol{E}2 = \varnothing$, то гиперграф является неориентированным, а если $\boldsymbol{E}2 = \boldsymbol{E}$

& $\boldsymbol{E}1 = \varnothing$, то гиперграф является (полностью) ориентированным. В зависимости от мощности множеств $\boldsymbol{E}1$ и $\boldsymbol{E}2$ вводятся соответствующие понятия. Если $|\boldsymbol{E}1| = |\boldsymbol{E}2|$, то гиперграф будет *ориентированно-неориентированным*. Если $|\boldsymbol{E}1| > |\boldsymbol{E}2|$, то гиперграф будет *почти неориентированным*. Если $|\boldsymbol{E}1| < |\boldsymbol{E}2|$, то гиперграф будет *почти ориентированным*.

Если элементам гиперграфа приписаны символы (или цепочки символов) из некоторого множества, то он является *раскрашенным гиперграфом* или гиперграфом с помеченными вершинами и ребрами. Цепочки символов – это имена понятий и отношений онтологии, представленной раскрашенным, частично ориентированным гиперграфом. Такой гиперграф будем называть *семантическим гиперграфом*.

3 Структура прикладной онтологии

Сегодня в теории принято классифицировать онтологии по степени зависимости от задач или прикладной области, по языку представления онтологических знаний и его выразительным возможностям и другим параметрам [6]. Прикладные онтологии описывают концепты, которые зависят как от онтологии задач, так и от онтологии предметной области. Онтологический инжиниринг при этом подразумевает: определение классов понятий в онтологии; наведение таксономии на классах; разработку структур понятий и ситуаций; определение свойств понятий и значений этих свойств; процедуры вывода и преобразования ситуаций.

Прикладная онтология «Планиметрия» разрабатывается на основе общих принципов построения онтологий, но с учетом использования в качестве языка представления знаний семантических гиперграфов. Данный формализм позволят определить онтологию в виде семантического гиперграфа:

$\mathbf{O}\ (\boldsymbol{X}, \boldsymbol{R}, \boldsymbol{I})$, (1)

где $\boldsymbol{X}$ – множество понятий проблемной среды (вершины гиперграфа), $\boldsymbol{R}$ – множество отношений между понятиями (дуги и ребра гиперграфа), а $\boldsymbol{I}$ – множество имен понятий и отношений в данной предметной области.

3.1 Определение понятий

Множество понятий проблемной среды разделяется на несколько подмножеств:

$\boldsymbol{X} = \{X1,..., Xk\}$, (2)

где *X1* – это класс всех подклассов проблемной среды (в онтологии «Планиметрия» это класс *Плоская Фигура*). *X2* – это подклассы «структурных» понятий проблемной среды (*Треугольник*, *Параллелограмм*, *Трапеция*, *Эллипс*, *Кольцо* и др.), *X3* – классы «составляющих» структурные понятия (*Сторона*, *Основание*, *Катет* и др.), *X4* – свойства понятий (*Периметр*, *Площадь*, *Длина* и т.д.), *X5* – значение свойств (значение угла αi, значение длины стороны lj и др.). На семантическом гиперграфе (также как и на семантической сети) можно представлять и результаты арифметических, логических и других операций, поэтому в онтологии можно вводить соответствующие вершины (например, *Разность*, *Сумма*, а также формулы подсчета периметра, площади и т.п.). Конечно, интерпретация подобных вершин вместе с соответствующими отношениями будет отличаться от интерпретации других вершин. Следует отметить, что множество понятий $\boldsymbol{X}$ является открытым – его можно расширять по мере необходимости.

Класс *Плоская Фигура*, охватывающий все планиметрические фигуры, имеет подклассы, которые представляют более конкретные понятия, чем надкласс. Каждый подкласс может иметь свои подклассы. Например, класс *Треугольник* разделяется на подклассы *Прямоугольный* и *Равнобедренный Треугольник*. А *Равносторонний Треугольник* является подклассом класса *Равнобедренный Треугольник*.

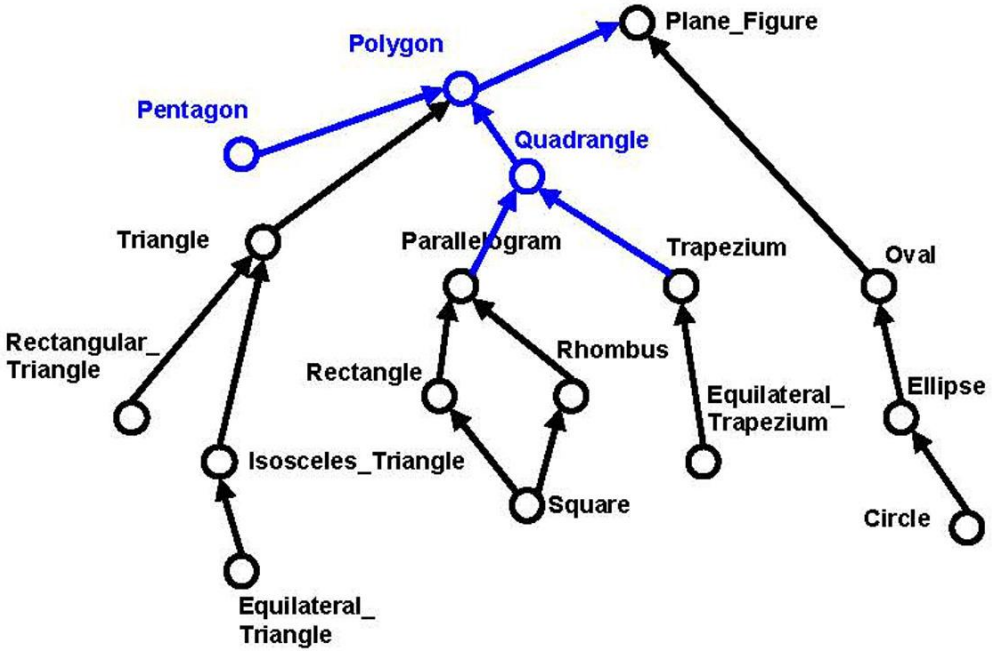

Рис.1. Таксономический фрагмент онтологии «Плоская геометрическая фигура»

Отношения между классами, подклассами и надклассами понятий организуются в виде **таксономии** или **таксономической иерархии**. Для представления таксономии используется отношение *является_видом* (*AKindOf*). Таксономический фрагмент онтологии, представляющий собой *2*-однородный ориентированный гиперграф, представлен на рис. 1.

3.2 Определение отношений

При разработке и при использовании любой онтологии необходимо определиться с перечнем используемых отношений. На сегодняшний день нет общепринятого полного перечня отношений за исключением десятка общезначимых отношений (например, *A_Kind_Of, part_of, have_value, have_structure* и др.). Тем не менее, всю совокупность отношений в онтологии стоит разделить на несколько подмножеств:

$\boldsymbol{R} = \{R1,\ldots, Rm\}$, (3)

где $R1$ – **общезначимые** отношения (см. выше), $R2$ – **арифметические**, $R3$ – **логические** (И, ИЛИ, НЕ и др.) и т.д. и Rk – **предметные** для данного приложения отношения. Общезначимые отношения (свойства некоторых общезначимых отношений представлены в [7]) носят в основном декларативный характер: они служат для нахождения путей поиска требуемых знаний. Арифметические, логические, функциональные и предметные отношения задаются как декларативно, так и операционально. Последнее означает, что с именем данного отношения связана соответствующая процедура, с помощью которой осуществляется интерпретация высказывания в реальной или «виртуальной» среде. Например, отношение *соприкасается_в_концевой_точке* предполагает, что существует процедура, с помощью которой на входном изображении осуществляется поиск двух отрезков прямых, соприкасающихся друг с другом концевыми точками. К предметным отношениям онтологии «Планиметрия» также относятся отношения: *образует, параллельно, перпендикулярно, вычисляется_по_формуле, совпадает, делит_пополам, исходит_из, является_внутренней_точкой* и т.д.

Отметим, что множество $\boldsymbol{R}$ является открытым – в него можно добавлять необходимые подмножества.

3.3 Определение имен понятий и отношений

Имена понятий и отношений онтологии выбираются на основе терминологии, соответствующей естественно-языковым реалиям в данной области, чтобы упростить процесс разработки онтологии для эксперта и инженера по знаниям. Но и здесь процесс кодификации далек от завершения, поэтому каждый разработчик предметной онтологии по возможности использует общепринятую терминологию (в основном для имен понятий) и собственные предпочтения – для имен отношений. В планиметрии существует устоявшаяся терминология для обозначения понятий (еще со времен Древней Греции). И понятия обозначаются цепочками символов очень похожими на слова естественного языка, например, либо английского, либо русского. С наименованием отношений – вопрос сложнее, поэтому и разнобоя больше.

Вообще процесс наименования определяется не только предметной областью и простотой разработки, но и вопросами хранения информации в памяти системы для различения однотипных понятий. Но этот вопрос при общем описании онтологии выходит за рамки данной работы. Для иллюстрации на рис. 1, 3 и 4 понятия и отношения представлены «словами» английского языка, а на рис. 2 - «словами» русского языка.

3.4 Структуры понятий и ситуаций

Один объект может составлять часть другого объекта – это привычно для описания окружающего нас мира. Глаза – это часть лица, Крыло или Кабина – часть Самолета, а данная статья – часть трудов конференции. Для отображения таких видов взаимоотношений целого и частей служат два вида понятий.

Составные понятия, в которых используется отношение *является_частью* (*Part_Of*). Для таких понятий бывает достаточным просто перечисление частей, например, оглавление для трудов конференции или фрагмент онтологии для понятия *Отрезок_Прямой* во взаимоотношении с понятиями *Полупрямая*, *Прямая* и видовыми концептами *Катет*, *Гипотенуза*, *Диагональ* и др.

Там, где важны еще и отношения между частями, служат **структурные понятия.** Для выделения структуры используется отношение *имеет_структуру (Have_Str)*. Понятие *Треугольник* в виде семантического гиперграфа представлено на рис. 2. Здесь структура выделяется гипердугой (выделена пунктиром), где *началом* дуги является вершина, помеченная словом *Треугольник*, а конечными вершинами будут все вершины, охваченные пунктирной областью. Отметим, что некоторые целостные свойства понятия могут быть представлены как внутри структуры этого понятия, так и в виде интегрального его свойства. В данном примере внутренним свойством является зависимость сторон треугольника. Внутри пунктирной области выделяется неориентированный подгиперграф с ребром, помеченным цепочкой символов F3 (формулой, определяющей взаимоотношения сторон треугольника). В целом гиперграф, изображенный на рис. 2, будет *почти ориентированным*, т.к. мощность ребер меньше мощности дуг, т.е. $|\boldsymbol{E}1| < |\boldsymbol{E}2|$.

Полная структурная часть понятия может оказаться не столь наглядной, как хотелось бы. Для этого предусматривается возможность при разработке и отладке онтологии представлять эти структуры в виде нескольких *стратов* – структур, в которых присутствуют базовые элементы, а в остальных частях определены разные компоненты. На рис.2 для наглядности понятие *Треугольник* определено в виде структуры трех отрезков прямых, соединенных концевыми точками. Хотя полная структура понятия *Треугольник* дополнительно включает биссектрисы, медианы, высоты, углы треугольника и другие элементы.

Ситуация – это такие описания, которые «компонуются» из некоторого множества взаимосвязанных определенным образом понятий. Они чаще всего не имеют статус понятия и носят временный характер. Такие описания возникают в процессе формирования результата на поступление входной информации. Иногда такого рода

описания (скажем, в зависимости от частоты их появления) могут оформляться в онтологии в качестве некоторого сложного понятия. Например, вписанная в треугольник окружность есть ситуация, полученная из описания треугольника, в который добавили описание окружности с определенными ограничениями (окружность с такими параметрами, при которых она касается трех сторон треугольника). Такая ситуация в планиметрии запомнена в качестве структурного понятия *Вписанная* (*в Треугольник*) *Окружность*.

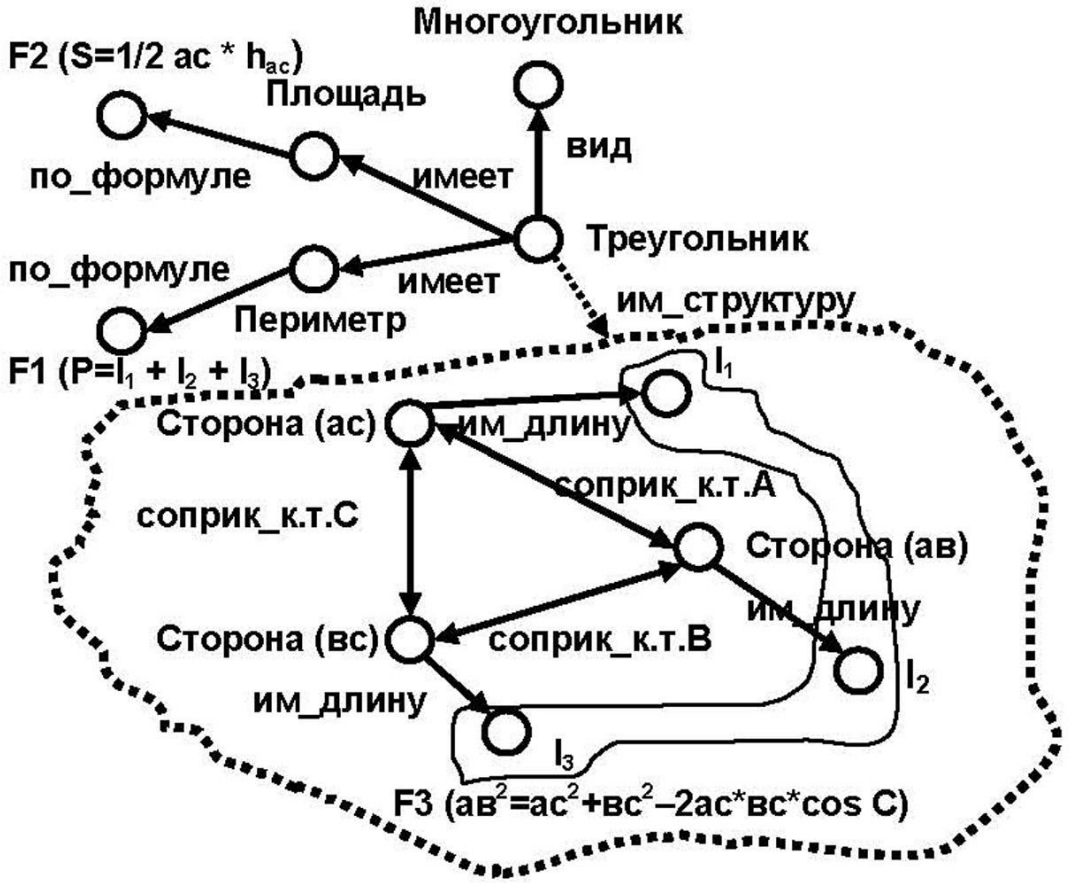

Рис.2. Фрагмент структуры понятия *Треугольник*

3.5 Определение свойств и значений свойств

Большинство понятий онтологии обладает соответствующими свойствами: для понятия *Треугольник* это может быть *Площадь*, *Периметр*, для понятия *Отрезок_Прямой – Длина* и т.д. Каждое свойство может быть задано конкретным значением или способом (например, формулой, реализуемой с помощью присоединенной процедуры) вычисления значения данного свойства. На рис. 3 интегральными свойствами понятия *Ромб* являются *Площадь* и *Периметр*. Значения этих свойств для экземпляра понятия подсчитываются по формулам F14 и F12, которые вычисляются процедурами при означенном (конкретном, числовом) значении внутреннего свойства понятия *Ромб*: длины стороны. В представлении на языке семантического гиперграфа длина стороны связывается с понятиями *Сторона* (*Side*) отношением: *имеет_длину* (*have_length*).

Из описаний структур понятий и ситуаций (см. рис. 2, 3, 4) можно заметить, что для каждой из подсистем интегральной системы информация об объектах избыточна. Например, для «отделения» (распознавания) треугольника от четырехугольника достаточно для распознаваемого объекта извлечь данные о количестве отрезков. При анализе некоторого естественно-языкового текста нет необходимости иметь в структуре

Треугольника описания медиан, биссектрис и других составляющих. Но если иметь в виду в совокупности все функции интегральной системы (не только распознавание, а и анализ, и синтез изображений, не только анализ естественно-языкового текста, но и его синтез, а также поиск решения задачи, ответы на вопросы и т.д.), то эта избыточность уже не кажется излишней.

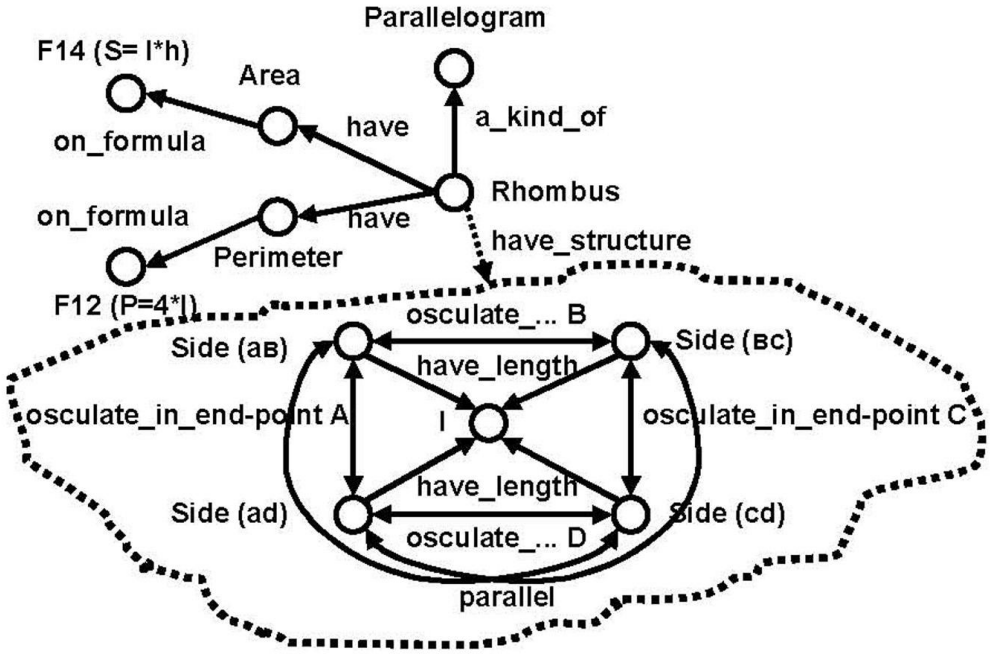

Рис.3. Фрагмент структуры понятия *Ромб*

4 Процедуры на прикладной онтологии

Использование онтологии в качестве модели предметной области определяется некоторым множеством процедур, которые кратко рассмотрены ниже.

Таксономический фрагмент онтологии необходим кроме всего прочего для использования информации о **наследовании**. Если известно, что все экземпляры класса *Многоугольник* имеют площадь, и утверждается, что *Треугольник* является подклассом класса *Многоугольник*, а *Равнобедренный Треугольник* является подклассом класса *Треугольник*, то каждый равнобедренный треугольник имеет площадь. Если объект принадлежит к нескольким классам, то в этом случае мы имеем **множественное наследование**. Таким примером в данной онтологии служит *Квадрат*, который одновременно принадлежит и к понятию *Прямоугольник* и к понятию *Ромб*.

Важной особенностью представления знаний в онтологии – возможность для понятий **задавать по умолчанию значения свойств**. Это позволяет естественным образом вводить исключения. Значение по умолчанию перекрывается более конкретным значением. Например, площадь любого треугольника подсчитывается по формуле 1/2 ac*hac (значение по умолчанию), а для прямоугольного треугольника площадь можно подсчитать по более простой формуле 1/2 ac*bc (более конкретное значение).

Возможно, излишне хранить информацию и о прямой, и об **обратной связи**. Тем не менее, с точки зрения приобретения и использования знаний в онтологии удобно иметь доступной эту информацию в явном виде. Например, связь *имеет_длину* является обратной по отношению к *является_длиной* или *входит_в_структуру* обратная к *имеет_в_структуре*. Система приобретения знаний может автоматически заполнить в описаниях и прямые и обратные дуги, обеспечивая полноту и согласованность базы знаний.

Выразительная мощность семантических гиперграфов проявляется, если ввести **присоединенные процедуры** – метод, с помощью которого осуществляется вызов специальной процедуры, предназначенной для обработки определенных понятий и отношений. Примером такой процедуры является упоминаемая выше процедура, связанная с отношением *вычисление_по_формуле* (*on_formula*).

При оперировании с конкретными экземплярами понятий реализуется процедура **означивания.** По входной информации все или некоторые переменные в описании понятия (включая и структуру) принимают конкретные значения. Некоторые переменные могут маркироваться знаком (например, знаком «?») – для них необходимо найти значение по условиям задачи. Например, если во входных данных относительно понятия *Равнобочная Трапеция* указаны длина одного из оснований и боковой стороны, то процедура означивания предполагает задание в обобщенной структуре этого понятия вместо переменных (длины основания и боковой стороны) их конкретных значений.

Фрагменты онтологии, включая структуры и свойства понятий, при решении конкретных задач становятся материалом для описания ситуации, которая определяется входными данными. В дополнение к этим фрагментам предполагается введение понятий и связей, определяемых входными условиями (например, текстом геометрической задачи). Это реализуется с помощью процедур **дополнения**. Рассмотрим данную процедуру (+ некоторые другие) на следующем примере. Пусть задан текст геометрической задачи:

Площадь треугольника, один из углов которого равен разности двух других, равна площади квадрата, сторона которого совпадает с одной из сторон этого треугольника. Найти углы данного треугольника.

В описании понятия *Треугольник* никакой информации о «разности двух углов» и о «равенстве этой разности третьему углу» быть не может. А ситуация, описывающая данный текст должна ее включать. В этом случае подключаются процедуры дополнения, которые на основе присутствия знания в онтологии о понятии *Разность* (*x, y*) и о связи *равно* модифицируют описание ситуации введением соответствующих узлов и дуг, как это представлено на рис. 4. Данный рисунок иллюстрирует процедуры дополнения для понятий *Треугольник* и *Квадрат* и процедуры частичного означивания. Выделенные узлы и жирные связи отображают те объекты, которые представлены в тексте задачи. Знак вопроса при значениях понятий *Угол* определяет то, что эти значения необходимо найти в процессе решения геометрической задачи.

При любом языке представления знаний необходимой процедурой является **сопоставление** описаний. Это либо сравнение по образцу (pattern matching), либо поиск изоморфизма гиперграфов. Последняя процедура, описанная в [8], предполагает сопоставление семантического представления геометрической фигуры или текста геометрической задачи со структурными фрагментами предметной онтологии. Результат подобного сопоставления представлен на рис. 4., где выделенные узлы сопоставлены с фрагментами семантического описания текста задачи: это понятия *Треугольник* (*Triangle*) и его *Площадь* (*Area*), углы треугольника, их значения и одна из сторон этого треугольника; понятия Квадрат (*Square*), его *Площадь* и *Сторона* (*Side*).

Сопоставление при поиске изоморфизма, вообще говоря, дает спектр результатов в зависимости от сопоставляемых подгиперграфов и от использования информации об иерархии понятий. То есть такая процедура дает возможность варьировать некоторыми параметрами при сопоставлении и получать разные результаты исходя из внешних критериев.

Процедуры поиска необходимой информации в онтологии для разных подсистем отличаются друг от друга. Например, при выделении на изображении *Отрезок_Прямой* (*Line_Segment*) путь поиска структуры-гипотезы об анализируемом объекте будет (см. рис.4):

Line_Segment → обратное отношение от *a_kind_of* → одно из множества {*Side, Median, Diagonal*} → *enter_structure* → одно из множества {*Triangle, Square*}.

А при анализе фрагмента текста «сторона треугольника» путь поиска будет:
Side → *enter_structure* → *Triangle*.

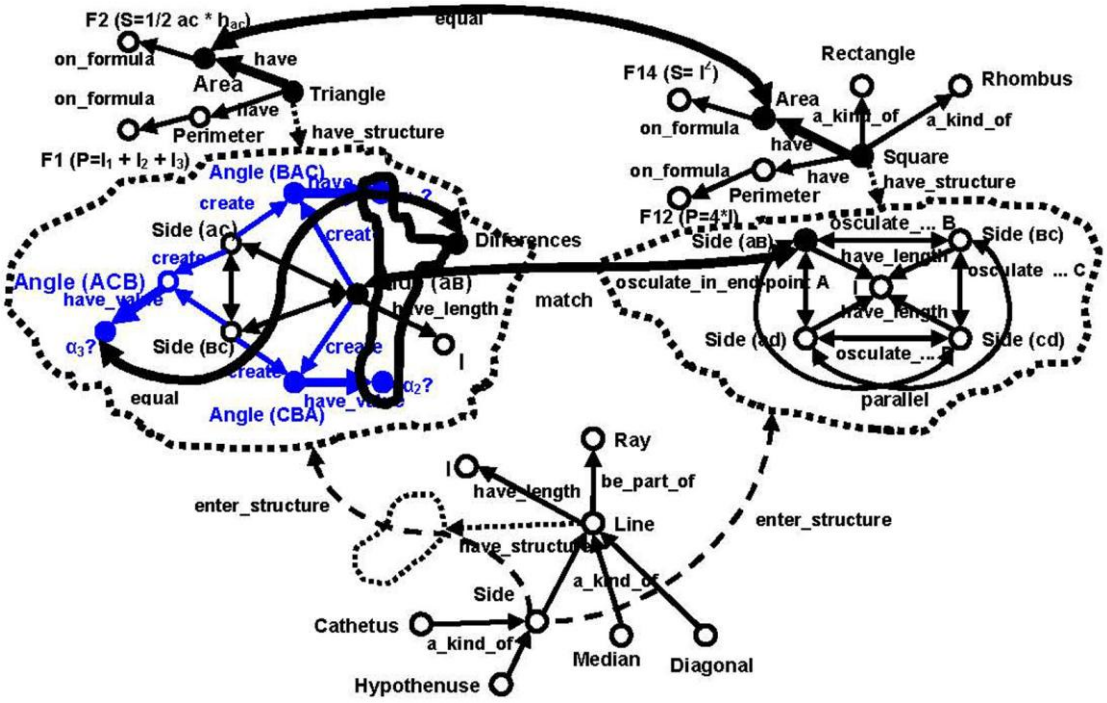

Рис.4. Частично означенная структура ситуации «Треугольник и Квадрат» для геометрической задачи

Такое различие определяется тем, что зрительная система оперирует своими понятиями (отрезками, дугами, точками и т.п.), а естественно-языковая – своими (сторонами, катетами, диагоналями и т.д.). И эти множества не обязательно всегда и полностью совпадают.

Процедуры верификации при отладке онтологии и, что очень важно, при автоматическом ее формировании требуют привлечения механизмов синтеза для анализирующих подсистем. Режимов верификации в интегральной системе может быть несколько. Например, если на входе системы задан текст геометрической задачи (см. выше), то система может синтезировать по этому тексту соответствующее изображение, представленное на рис. 5.

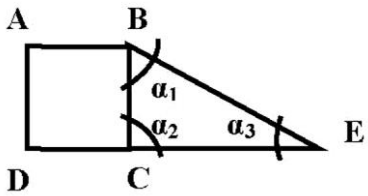

Рис.5. Изображение фигуры, иллюстрирующей текст геометрической задачи

Отметим, что если изображение рис. 5 подать на вход зрительной системы (без сопровождающего текста), то решить ей, что это за «геометрическая ситуация» будет трудно. С точки зрения «изолированного» анализа здесь есть неопределенность: то ли это квадрат и треугольник, имеющий общую сторону с квадратом, то ли это трапеция с высотой, опущенной из точки B. Если бы анализ изображения и текста происходил параллельно, то эта неопределенность могла бы быть разрешена самой системой.

5 Заключение

Динамика развития отдельных систем и методов искусственного интеллекта позволяет предполагать, что получить новый качественный результат в этой области можно только на уровне комплексирования подсистем.

В первую очередь это касается «связки» зрения и естественного языка (ЕЯ). Зрительная система без ЕЯ всего лишь «распознаватель». Это то, чем обладают и животные и люди. И только тандем «ЕЯ + зрение» «поднимает» зрительную систему до уровня «понимающего» аппарата, а системе обработки ЕЯ позволяет разрешать многие неопределенности и текста и диалога.

Литература

[1] The First Conference on Artificial General Intelligence / P. Wang et al. (Eds.), AGI-08, 1-3 March, 2008, Memphis, USA. IOS Press, 2008.

[2] Хахалин Г.К., Воскресенский А.Л. Мультизадачное использование прикладной онтологии // Труды XI национальной конференции по Искусственному Интеллекту с международным участием – КИИ-2008. М.: URSS, 2008, т.1, С. 112-123.

[3] Башмаков И.А., Башмаков И.А. Интеллектуальные информационные технологии. – М.: Изд-во МГТУ им. Н.Э. Баумана, 2006, 302 с.

[4] ЗыковА.А. Гиперграфы // Успехи математических наук. 1974, Т. 29, вып. 6, С. 89–154.

[5] Визинг В.Г. О раскраске инциденторов в гиперграфе // Дискретный анализ и исследование операций, июль—сентябрь 2007, Серия 1, Том 14, № 3, С. 40–45.

[6] Гладун А.Я., Рогушина Ю.В.. Онтологии в корпоративных системах. // Корпоративные системы, №1, 2006.

[7] Боровикова О.И., Загорулько Ю.А. Подход к представлению знаний в многоязычных информационных системах // Труды XI национальной конференции по Искусственному Интеллекту с международным участием – КИИ-2008. М.: URSS, 2008, т. 3, С. 155-163.

[8] Хахалин Г.К. Использование гиперграфов в лингвистической трансляции // Труды Международного семинара "Диалог'99" по компьютерной лингвистике и ее приложениям. – М., 1999, С. 315-320.

ЗАКЛЮЧЕНИЕ

Представленная **антология** развивающейся дисциплины **онтологии** показывает, насколько близки её научные результаты к практическому применению во многих областях человеческой деятельности. Информационный вал, вызванный повсеместным использованием и применением компьютерной техники, интернета, многочисленных баз данных и различных систем CAD/CAM/CAE, определил необходимость и важность:

- упорядочения, структурирования и классификации всего и вся: объектов, связей, процессов, в том числе и проектирования;
- четкого и однозначного понимания и толкования циркулирующих понятий и сущностей.

Без онтологического анализа добиться этого не возможно. Так же как и составить полное представление об указанной проблеме лишь по прочтении этой брошюры. Так что дерзайте, благо пытливым и ищущим всегда сопутствует удача!