OSTIS-2014
(Open Semantic Technologies for Intelligent Systems)

УДК 004.822:514

ПОСТРОЕНИЕ ИНТЕЛЛЕКТУАЛЬНЫХ СИСТЕМ, ОСНОВАННЫХ НА ЗНАНИЯХ, С ПОВТОРНЫМ ИСПОЛЬЗОВАНИЕМ КОМПОНЕНТОВ

Борисов А.Н.

Рижский технический университет, г. Рига, Латвия
Arkadijs.Borisovs@cs.rtu.lv

В работе рассмотрены теоретические основы и средства построения интеллектуальных систем с многократным использованием компонентов на основе методов инженерной онтологии. Приведены два основных семейства инструментальных средств: модельно-ориентированный подход и подход, использующий декомпозицию решаемой проблемы. Описаны основные языки представления онтологий, а также программные средства построения онтологий. Рассмотрены средства представления декларативных знаний и механизмы обработки процедурных знаний, а также средства взаимодействия онтологических баз знаний и декларативных правил. Проанализированы дискуссионные вопросы развития реляционных языков и онтологической семантики.
Ключевые слова: инженерная онтология; повторное использование компонентов; системы, основанные на знаниях.

Введение

Традиционные системы, основанные на знаниях, разработанные в 70-80гг. при их реальном использовании встретили много проблем, показывающих их несовершенство.

Введение технологии «пустых» систем, основанных на знаниях, лишь частично сняло возникшие проблемы. Появилась идея повторного использования компонент при проектировании систем. Этот подход требует декомпозиции задачи на подзадачи, а именно - декомпозиции предметной области, декомпозиции проблемы. Каждая из частных полученных после декомпозиции подзадач требует своей порции знаний, которые позволят решать эти частные подзадачи. В этом случае понятие онтологии используется для описания таких частных подзадач.

1. Ранние системы, основанные на знаниях (70-80гг.)

Создатели первых систем, основанных на знаниях [Jackson, 1999], ожидали, что их системы будут идеально решать задачу повторно используемых компонентов [Болотова, 2012]. Эти системы состояли из двух частей:

- базы знаний, содержавшей продукционные правила, то есть декларативную часть;
- блока логического вывода, обеспечивавшего процедурную функцию. Предполагалось, что данный подход может быть применѐн к различным решаемым задачам. Однако эти надежды не оправдались по следующим причинам:
- создание базы продукционных правил для новой предметной области оказалось весьма дорогостоящим;
- проблема «узкого горлышка» не может быть решена, так как существуют трудности описания предметной области в терминах понятий низкого уровня абстракции, необходимых для функционирования блока логического вывода.

2. Современные системы, основанные на знаниях

Стало очевидным, что экспертные знания, представленные на уровне продукционных правил или фреймов не обеспечивают достаточно абстрактную модель для создания интеллектуальных систем [Russel, 2010].

С целью абстрагирования от конкретной задачи и перехода к мета-описанию поведения системы предложены четыре основных подхода:

- понятие родовой, общей (generic) задачи, например, использование предметно независимого метода решения задачи;
- понятие ограниченной роли метода решения задачи, которое позволяет рассматривать, например, различные роли составляющих метода решения и, соответственно, определять порцию знаний для конкретной прикладной задачи;

- предположение об «открытости мира»;
- библиотека онтологического описания знаний.

3. Теоретические основы методов многократного использования компонентов

3.1. Предположение «открытого мира»

Онтология, реализованная на языке OWL (Web Ontology Language), представляет собой множество декларативных утверждений о сущностях словаря предметной области [Лапшин, 2010; Hepp, 2007]. OWL предполагает концепцию «открытого мира», в соответствии с которой применимость описаний предметной области, помещѐнных в конкретном физическом документе, не ограничивается лишь рамками этого документа – содержание онтологии может быть использовано и дополнено другими документами, добавляющими новые факты о тех же сущностях или описывающими другую предметную область в терминах данной. «Открытость мира» достигается путѐм добавления URI каждому элементу онтологии, что позволяет воспринимать описанную на OWL онтологию как часть всеобщего объединѐнного знания.

3.2. Библиотеки методов решения проблем

Библиотека Бенджамина [Gómez-Pérez, 1999] содержит набор методов решения задач (Problem Solving Methods-PSMs). Каждый PSM декомпозирует задачу на подзадачи и/или примитивные логические заключения и располагает структурой для контролирования выполнения подзадач и примитивных заключений, а также – для получения общего решения задачи. Подзадача – это задача, для решения которой может быть применен другой подходящий метод. Каждая задача имеет описание информации на входе и на выходе. Каждый PSM имеет множество критериев пригодности, которые используются для выбора метода в приложении. Эти критерии выражаются в терминах свойств среды, предметной области и имеющихся знаний; в методологии CommonKADS (Knowledge Acquisition and Documentation Structuring) все это называется свойствами задачи.

3.3. Общая задача

Считается, что онтология задачи является общей (generic), если она независима от специфики предметной области, или приложения, или метода вывода [Chandrasekaran, 1998]. Например, в проблеме планирования действий онтология задачи формализуется без указания любой специфики задачи планирования, но в то же время извлекает и объединяет важные концептуальные теоретические отличия, которые существуют в большинстве парадигм планирования. Понятие общей задачи может быть распространено на метод решения, на онтологию предметной области и является ключевым подходом для построения библиотеки повторно используемых компонентов в процессе проектирования систем, основанных на знаниях.

3.4. Методы ограничения роли (Role-limiting methods – RLM)

Подход, использующий RLM, можно охарактеризовать как оболочный. Такая оболочка получается вместе с реализацией конкретного PSM, таким образом, еѐ можно использовать только для решения такого типа задач, для которого PSM подходит. Данный PSM также определяет роли, которые знания могут играть в ходе процесса решения задач; он фиксирует представление знаний для ролей, так что эксперту необходимо подтвердить общие понятия и отношения, которые определяются с помощью этих ролей [Mizoguchi, 1995]. RLM, основанные на эвристической классификации, например, предлагают роль, точно определяемую экспертом. Используя эту роль, эксперт:

- должен указать, какое характерное для предметной области понятие соответствует этой роли, например, «данные о пациенте»;
- должен обеспечить экземпляры предметной области для этого понятия, например, конкретные факты о пациентах. Важно подчеркнуть, что тип знаний, который использует RLM, определен заранее. Следовательно, получение требуемых экземпляров, характерных для предметной области, может поддерживаться графическими интерфейсами, которые выполнены специально для данного PSM.

4. Инструментальные средства для построения систем, основанных на знаниях

4.1. Модельно-ориентированный подход

Исходной мотивацией для модели экспертизы KADS послужила поддержка вербальной информации от человека-эксперта, в частности, рассуждения вслух. Модель экспертизы это реализация независимой модели экспертных знаний о решении проблемы [Gaševic, 2009]. Еѐ можно разделить на четыре уровня по различным видам знаний с ограниченным взаимодействием между уровнями:

- *уровень предметной области,* который описывает концепты и отношения между ними;
- *уровень вывода,* который описывает элементарные выводы, которые можно применить к знаниям о предметной области, называемые источниками знаний;
- *уровень задачи,* со структурой управления, называемой структурой задачи, которая показывает, как можно комбинировать источники знаний, чтобы достичь цели;
- *стратегический уровень,* содержит знания о том, как создавать структуры задачи и находить решения в тупиковых случаях.

Стратегический уровень обычно явно не

моделируется, в частности потому, что суждения о стратегии не кажутся важными при решении рутинной задачи, которая является объектом при разработке системы. В системе, основанной на знаниях, стратегический уровень отражает способность пользователя вникнуть в процесс решения задачи.

Так же как и технологии Protégé, KADS со временем развивается. Методология CommonKADS создавалась с целью использования более точного, формализованного языка для описания моделей экспертизы, а также - путѐм включения новых идей относительно компонентов экспертизы. CommonKADS также ввѐла идею библиотеки повторного использования методов решения проблем, построенной на базе моделей уровня вывода и уровня задач KADS.

4.2. Подход, основанный на декомпозиции задачи

С момента представления системы Protégé были введены ещѐ три версии (Protégé -I, Protégé -II, and Protégé/Win), пока не появилась текущая версия Protégé -2000 [Gómez-Pérez, 2004].

Современные версии Protégé написаны на языке Java, который позволяет создавать не привязанные к определѐнной операционной системе приложения. Т.е. если в целевой операционной системе установлена Java Machine – JDK нужной версии, то в ней можно использовать и Protégé версии выше 3.0.

Модель знаний Protégé -2000 основана на модели Open Knowledge Base Connectivity (OKBC), которая обеспечивает единообразную модель Knowledge Representation Systems (KRS) и является более гибкой, чем другие модели знаний, использованные в более ранних версиях Protégé. Знания сохраняются в специализированном формате. Это позволяет читать и писать в PDF файлы и использовать формат реляционных баз данных. Разработка инструментальной среды Protégé/Win изначально вызвана прагматическими соображениями: Protégé-II могла работать только под операционной системой NeXTStep, и чтобы расширить базу пользователей, необходимо было так переработать систему, чтобы она могла работать под операционной системой Windows. Однако самым главным вкладом Protégé/Win является расширение коллектива внешних пользователей. Protégé/Win свободно доступна как легко инсталлируемое приложение для любого пользователя. Впервые пользователь начал получать обратную связь от реального мира о проблемах и особенностях прикладной системы. Эта связь повлияла на развитие системы. Protégé-2000 позволяет пользователям вносить изменения в онтологию и предоставляет undo/redu функциональность. Более того, даѐтся возможность архивировать различные версии онтологии и вернуться к предыдущим версиям. Это один из немногих инструментов, которые поддерживают работу с версиями онтологии.

4.3. Подход, основанный на WWW. Проект IBROW3

Основная цель проекта IBROW3(An Intelligent Brokering Service for Knowledge Component Reuse on the World-Wide Web) заключалась в разработке архитектуры, которая облегчала «интеллектуальный брокерский сервис» для создания системы, основанной на знаниях (Knowledge-based System – KBS), с помощью многократного использования компонентов знаний третьей стороны через WWW. Мета-онтология UPML (United Problem-Solving Method Development Language) была разработана для поддержки определения компонентов знаний. UPML вводит четырехкомпонентную архитектуру для систем знаний посредством расширения модели экспертизы CommonKADS. Идентифицируются следующие компоненты:

- *Задача.* Определяет проблему, которая будет решаться.
- *Метод решения задачи*. Описывает процесс вывода заключений.
- *Модель предметной области.* Определяет знания о предметной области.
- *Адаптер.* Облегчает соединение трех других компонент, определенных с помощью независимых и, следовательно, многократно используемых спецификаций.

5. Средства построения KBS на основе многократно используемых компонентов

5.1. Языки для описания онтологий

Существует семейство языков представления онтологий RuleML, включающее DAML+OIL, RDF & RDF Schema, OWL и пр. Рассмотрим некоторые из них.

5.1.1. *RDF (Resource Description Framework)*

RDF (Resource Description Framework) представляет собой технологическое решение для представления информации в WWW. Любое выражение RDF основано на коллекции так называемых триплетов, каждый из которых состоит из субъекта (subject), предиката (predicate) и объекта (object). Триплет представляет собой утверждение о связи между понятиями «субъект» и «объект», обозначенными как узлы, которые эта связь соединяет. Направление связи важно: оно всегда указывает на объект. Таким образом, RDF представляет модель (framework) для аннотации ресурсов Сети при помощи XML-синтаксиса и триплетной модели (Subject, Predicate, Object), которая имеет наглядное графическое представление (ориентированный размеченный граф).

5.1.2. *RDF Schema*

RDF Schema является языком описания пользовательских словарей (наборов понятий) при помощи стандартизованного набора тегов в формате RDF-триплетов, который и составляет основу RDF Schema. Для простоты можно считать, что RDF

Schema предоставляет систему типов для RDF. Сам по себе RDF не включает никаких механизмов ни для описания свойств, ни для описания отношений между этими свойствами и другими ресурсами. Эту роль играет RDF Schema, который определяет понятия «класс», «свойство» и некоторые другие, позволяющие описывать классы, свойства и другие ресурсы.

5.1.3. *OWL – язык описания онтологий.*

OWL является развитием DAML + OIL, и был разработан для использования приложениями, которым требуется обрабатывать информацию, а не просто предоставлять еѐ агентам [Kamel, 2007]. OWL предлагает большую способность к взаимодействию, чем RDF(S), благодаря наличию дополнительного словаря вместе с поддержкой формальной семантики. У языка OWL есть три (по возрастанию выразительной мощности) подмножества: OWL Lite, OWL DL и OWL Full. OWL Lite содержит основные конструкции для описания классов и задания простых ограничений, OWL DL предлагает максимум возможностей, оставаясь при этом в рамках дескриптивной логики. OWL Full не гарантирует возможности существования вычислительной поддержки и рассчитан на пользователей, которые хотят получить максимальную выразительную мощность и свободу от RDF. В современных версиях стандарта OWL Lite отсутствует.

6. Среды для построения онтологий

Существует большой набор сред для поддержания онтологий: OilEd, Apollo, RDFedt, OntoLingua, OntoEdit, WebODE. Рассмотрим некоторые из них.

6.1. RDFedt

RDFedt даѐт возможность пользователю строить сложные и структурированные RDF and RSS документы. Он обеспечивает обзор сложных структур данных с элементами дерева и позволяет пользователю тестировать данные, делать комментарии и выдавать сообщения об ошибках с помощью дополнительных функций. RDFedt поддерживает основные элементы RDF, RDFS, и Dublin. RSS 1.0 обеспечивает такие модули как агрегирование, нотация, организация, изменение страницы, поточность и т.д. RSS 0.91 поддерживает декларирование и уровни стилей в XML, множества импортируемых элементов и автоматическое генерирование связанного списка на базе RDF из HTML (hypertext markup language) документа. RDFedt представляет собой текстовый редактор языков. Это не программа Java, этот редактор не является платформно-независимым, он работает только под Windows.

6.2. OntoLingua

OntoLingua разработана в лаборатории систем, основанных на знаниях, Стэнфордского университета. Эта библиотека обеспечивает среду сотрудничества для распределенных пользователей, набор инструментов для авторинга онтологий и библиотеку модульных, многократно используемых онтологий. Она также поддерживает (WWW) интерфейс и перевод в различные форматы. OntoLingua представляет собой библиотеку онтологий и сервер, доступ к которым можно получить через традиционный Web браузер. С использованием модуля Chimaera реорганизуется таксономия и решаются конфликты имѐн в базе знаний. Если пользователей несколько, они могут использовать библиотеку OntoLingua через блокировку опции *только для записи* и назначение уровня доступа пользователя.

7. Механизмы обработки процедурных правил

Существует набор средств обработки процедурных знаний: JENA, Pellet, JESS, RACER, OO jDREW.

7.1. Jess

Jess является аббревиатурой названия Java Expert System Shell. Jess представляет собой механизм правил и среду сценариев, написанные целиком на языке Java. Разработка Jess первоначально вдохновлялась оболочкой экспертной системы CLIPS, но затем переросла в законченную самостоятельную среду, на которую повлияла Java. Используя Jess, можно создавать Java-апплеты и приложения, которые обладают способностью "делать заключения", используя знания, которые представлены в форме декларативных правил. Так же как и CLIPS, Jess имеет подобный Lisp синтаксис, который обозначает или «Обработка списков» или «Множества избыточных скобок» в зависимости от того, кому адресуется этот вопрос.

7.2. Jena

Jena является фреймворком для семантических Web приложений, который включает API (Application Programming Interface) для RDF. Он поддерживает постоянное хранение, механизм запросов SPARQL и основанный на правилах механизм вывода для RDFS и OWL. Jena обеспечивает прямой и обратный вывод и гибридную модель выполнения вывода, а также - способы комбинирования вывода для RDFS/OWL с выводами на созданных правилах.

7.3. Pellet

Pellet это общедоступная машина вывода на базе Java, которая поддерживает вывод на уровне OWL-DL. Она представляет собой удобное средство для реализации тщательного и полного OWL DL вывода. Pellet обеспечивает такие функции вывода, как проверка на непротиворечивость, выполнимость, классификация и реализация понятий. Pellet включает также оптимизированный

механизм запросов, способный отвечать на запросы Abox.

8. Языки логического вывода на основе информации, представленной онтологиями

8.1. RuleML Initiative

RuleML Initiative разрабатывает открытый язык, основанный на правилах, для обработки RDF онтологий, что обеспечивает обмен правилами между различными системами включая распределенные программные компоненты на Web, неоднородные клиент-сервер системы, которые встречаются в больших корпорациях, и т.д. Язык RuleML использует XML-синтаксис для представления знаний в виде правил, а также - для организации взаимодействия между основными коммерческими и некоммерческими системами правил. RuleML строит иерархию подъязыков правил на базе XML, RDF, и OWL, например, SWRL.

8.2. SWRL (Semantic Web Rule Language)

SWRL (Semantic Web Rule Language) был предложен для того, чтобы увеличить возможности онтологии на основе OWL. SWRL правила обеспечивают процедурные знания, которые компенсируют некоторые ограничения логического вывода на базе онтологий, в частности, в плане идентификации семантических отношений между экземплярами. SWRL использует типичное логическое выражение антецедент $\rightarrow$ консеквент для представления семантических правил. Как антецедент, так и консеквент могут быть конъюнкцией одного или более атомов.

9. Средства, поддерживающие взаимодействие онтологий и правил

9.1. Protégé OWL плагин

Машина вывода Jess функционирует в рамках среды Protégé и является основой для интеграционной модели плагина JessTab. Так как Protégé и Jess выполнены при помощи языка Java, их можно запускать вместе в единой виртуальной машине Java. Этот подход даѐт возможность использовать Jess как интерактивное средство для действий с онтологиями и базами данных Protégé. Более того, становится возможным распространить процедуру изменений в Protégé на оболочку Jess.

9.2. SWRL JESS Tab

Выполнение правил SWRL требует наличия машины обработки правил, которая осуществляет вывод, используя в качестве входной информации множество правил и фактов. Любые новые выводимые факты используются как входные данные с тем, чтобы активировать по возможности больше правил (т.е., прямой вывод). В большинстве реализаций используется машина обработки правил Jess для того, чтобы сделать возможным SWRL вывод на множестве данных Protégé [Eriksson, 1995]. SWRL-правила хранятся как OWL экземпляры вместе с ассоциируемой с ними базой знаний. Прежде всего, SWRL-правила переводятся в Jess правила, используя SWRL JessBridge, который встроен в Protégé и добавлен в машину обработки правил Jess; затем онтологии и база знаний транслируются в Jess факты а также – вводятся в машину вывода; в-третьих, машина обработки правил Jess делает возможным вывод и производит результаты в формате Jess и, наконец, эти результаты переводятся обратно в формат OWL. SWRL-правила можно легко моделировать, используя механизм SWRL Jess Tab, который включен в Protégé и обеспечивает графический интерфейс для взаимодействия с механизмом SWRL JessBridge. Механизм SWRL Jess Tab позволяет вводить, удалять и редактировать SWRL-правила.

10. Вопросы для обсуждения

10.1. Реляционные языки в онтологии

Ситуационные модели управления одними из первых начали осуществлять формализацию объекта на языке, близком к естественному [Поспелов, 1986]. Основными сохраняемыми знаками являлись знаки двух видов: *понятия* и *отношения*. Главные идеи языка ситуационного управления основаны на работах Мартынова В.В. об универсальном семантическом коде (УСК) [Мартынов, 1997].

Простой ядерной конструкцией в универсальном семантическом коде является тройка вида (SAO). В этой тройке S соответствует субъекту, совершающему акцию, а O – объекту, на который направлена данная акция A. Такой тройке может, например, соответствовать предложение «Кран разгружает судно». В УСК вводится замкнутая система операций, позволяющих из простых ядерных конструкций строить более сложные цепочки.

Дальнейшее развитие реляционных языков связано с разработкой RDF представления, которое выражает семантические отношения на уровне примеров (экземпляров), выраженные в терминах тройки. В то же время, RDF Schema выражает отношения на уровне классов, описывая приемлемые отношения на уровне примеров (экземпляров), выраженные в терминах тройки. RDFS является мета-уровнем или более абстрактной моделью, описывающей уровень объектов в RDF. На сегодняшний день RDFS и OWL полностью удовлетворяют запросы онтологических систем как в концептуальном плане, так и в плане поддерживающих средств.

Совершенно естественно возникает вопрос о дальнейших путях развития реляционных языков.

10.2. Онтологическая семантика

В логике высказываний и в логике предикатов семантика операций определяется с помощью интерпретаций смысла. В существующей интерпретации смысла в данном случае семантика полностью удовлетворяет потребностям практики [Рубашкин, 2013]. В системах представления знаний, использующих семантические сети, можно также воспользоваться интерпретациями, принятыми в логике предикатов. Как известно, графическая интерпретация логики предикатов может быть осуществлена на основе объединения концептуальных графов и семантической сети. Таким образом, при смысловой интерпретации семантических сетей можно воспользоваться опытом, полученным при решении подобной задачи в логике предикатов.

Язык RDF, обладая высокой степенью общности, не позволяет варьировать глубиной и точностью семантических описаний. На решение этой задачи нацелен другой язык системы, OWL. В составе OWL продублированы средства формирования словаря семантических примитивов, разработанные для RDF. Тем самым обеспечена системная совместимость языков. Наряду с этим OWL позволяет уточнить смысл того или иного семантического примитива прежде всего за счет внесения ограничений на кардинальность отдельных классов и на использование отношений между классами. В этом случае при внесении ограничений на кардинальность в дескриптивной логике нарушается предположение об открытости мира. Так как в замкнутом мире действуют законы логики предикатов, здесь также можно воспользоваться методами интерпретации, принятыми в логике предикатов.

В общей системе RDF/OWL язык OWL играет роль инструмента для задания аксиоматики словаря семантических примитивов, в то время как RDF ориентирован, прежде всего, на семантическое представление конкретных фактов в рамках заданной на языке OWL аксиоматики. Системой RDF/OWL оформлена новая технология описания и представления семантики данных – технология Semantic Web.

Каковы альтернативные пути семантической интерпретации онтологии для открытых миров?

Заключение

Привлечение методов и средств инженерной онтологии стимулировало интенсивное развитие систем, основанных на знаниях, нового поколения. Одним из новых направлений является проектирование систем с повторным использованием компонентов. Разработаны фундаментальные теоретические основы этого подхода, однако для их успешной реализации на практике предстоит еще проделать большую работу.

Библиографический список

[Мартынов, 1997] Мартынов В.В. Универсальный семантический код / В.В. Мартынов; – Минск: Наука и техника, 1997.

[Поспелов, 1986] Поспелов Д.А. Ситуационное управление. Теория и практика / Д.А. Поспелов; – М.: Наука, 1986.

[Болотова, 2012] Болотова Л.С. Системы искусственного интеллекта: модели и технологии, основанные на знаниях / Л.С. Болотова; – М.: Финансы и статистика, 2012.

[Лапшин, 2010] Лапшин В.А. Онтологии в компьютерных системах / В.А. Лапшин; – М.: Научный мир, 2010.

[Рубашкин, 2013] Рубашкин В.Ш. Онтологическая семантика / В.Ш. Рубашкин; – М.: Физматлит, 2013.

[Chandrasekaran, 1998] Chandrasekaran B. Ontology of Tasks and Methods / B.Chandrasekaran, J.R.Josephson, V.R. Benjamins // IEEE Intelligent Systems, 14(1), 1998, P. 20-26.

[Eriksson, 1995] Eriksson H. Task modeling with reusable problem-solving methods / H. Eriksson // Artificial Intelligence. - Vol. 79, Issue 2. - 1995, P. 293–326.

[Gaševic, 2009] Gaševic D. Model Driven Engineering and Ontology Development / D. Gaševic; – Springer-Verlag, 2013.

[Gómez-Pérez, 2004] Gómez-Pérez A. Ontological Engineering / A. Gómez-Pérez, M. Fernández-López, O. Corcho; – Springer-Verlag, 2004.

[Hepp, 2007] Hepp M. Ontologies: State of the Art, Business Potential, and Grand Challenges, Chapter 1 / M. Hepp // Ontology Management: Semantic WEB, Semantic WEB Services, and Business Application, 2007, Springer, P. 3-22.

[Kamel, 2007] Kamel M.N. A methodology for developing ontologies using the Ontology Web Language (OWL) / M.N. Kamel // ICEIS 2007 - Proceedings of the Ninth International Conference on Enterprise Information Systems, June 12-16, Funchal, Madeira, Portugal, 2007, Springer, P. 261-268.

[Mizoguchi, 1995] Mizoguchi R., Vanwelkenhuysen J, Ikeda M. Task ontology for reuse of problem solving knowledge / R. Mizoguchi, J. Vanwelkenhuysen, M. Ikeda // Proceedings of the Second International Conference on Building and Sharing of Very Large-Scale Knowledge Bases (KB & KS'95), 1995, P. 45-59.

[Gómez-Pérez, 1999] Gómez-Pérez A., Benjamins V.R. Applications of ontologies and problem-solving methods / A. Gómez-Pérez, V.R. Benjamins // AI Magazine. - Vol. 20, N 1. - 1999, P. 119–122.

[Russel, 2010] Russel S., Norvig P. Artificial Intelligence: A Modern Approach. 3rd Edition / S. Russel, P. Norvig; – Prentice Hall, 2010.

[Jackson, 1999] Jackson P. Introduction to Expert Systems. 3rd Edition / P. Jackson; – Addison-Wesley, 1999.

ONTOLOGY-BASED INTELLIGENT SYSTEM CONSTRUCTION THROUGH COMPONENT REUSE

Borisov A.N. *

**Riga Technical University, Riga, Latvia*

Arkadijs.Borisovs@cs.rtu.lv

This work considers the theoretical foundations and tools for ontology-based intelligent system construction through component reuse. Two major tool families are discussed: the model-driven approach and the technique using the decomposition of the problem to be solved. Main ontology representation languages as well as ontology construction software are described. Declarative knowledge representation tools and mechanisms for processing procedural knowledge are examined as well as ontology knowledge base and declarative rule cooperation tools. Discussion questions addressing the development of relational languages and ontology semantics are analysed.